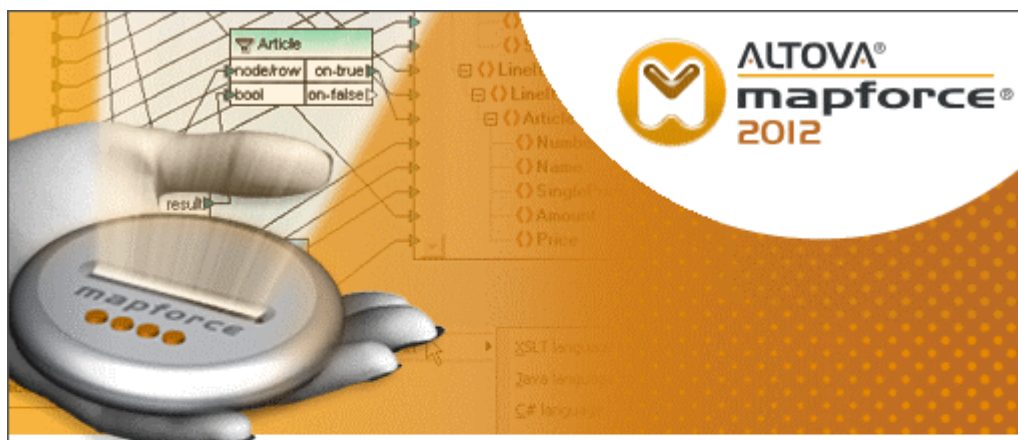


Benutzer- und Referenzhandbuch



Copyright © 1998–2012, Altova GmbH. All rights reserved. Use of this software is governed by and subject to an Altova software license agreement. XMLSpy, MapForce, StyleVision, SemanticWorks, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, AltovaXML, MissionKit, and ALTOVA as well as their logos are trademarks and/or registered trademarks of Altova GmbH. Patent pending.

ALTOVA®

XML, XSL, XHTML, and W3C are trademarks (registered in numerous countries) of the World Wide Web Consortium; marks of the W3C are registered and held by its host institutions, MIT, INRIA, and Keio. UNICODE and the Unicode Logo are trademarks of Unicode Inc. This software contains 3rd party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html

Altova MapForce 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

1	MapForce 2012	3
2	Was ist neu...	6
3	Übersicht über MapForce	10
3.1	Terminologie	12
4	MapForce Tutorial	18
4.1	Einrichten der Mapping-Umgebung	20
4.1.1	Hinzufügen von Komponenten zum Mapping-Fenster	21
4.2	Erstellen eines Mappings	24
4.2.1	Mappen von Schema-Elementen	25
4.2.2	Mappen von Daten mit Hilfe von Funktionen	28
4.2.3	Filtern von Daten	32
4.3	Generieren von XSLT 1.0- und 2.0-Code	36
4.4	Arbeiten mit mehreren Zielschemas / -dokumenten	37
4.4.1	Erstellen einer zweiten Zielschemakomponente	38
4.4.2	Anzeigen und Generieren mehrerer Zielschema-Ausgaben	42
4.5	Mappen mehrerer Quellelemente auf ein einziges Zielelement	44
4.5.1	Erzeugen der Mappings	46
4.5.2	Duplizieren von Input-Datenelementen	49
4.6	Input aus mehreren Dateien / Output in mehrere Dateien	54
4.6.1	Verarbeiten mehrerer Dateien pro Input-/Output-Komponente	55
4.7	Mappen einer Datenbank auf ein Schema	59
4.7.1	Herstellen einer Verbindung zu einer Access-Datenbank	61
4.7.2	Mappen von Datenbankdaten	63
5	MapForce Benutzeroberfläche	68
5.1	Das Register "Bibliotheken"	72
5.2	Das Register "Projekt"	74

5.3	Mapping-Fenster	78
5.4	Die Fenster XSLT/XSLT2/XQuery	80
5.5	Fenster "DB-Abfrage"	81
5.5.1	Browser-Fenster	82
5.5.2	SQL Editor	84
5.5.3	Register "Ergebnisse"	85
5.5.4	Register "Meldungen"	86
5.6	Fenster "Ausgabe" / BUILTIN-Ausführungsprozessor	88
5.7	Fenster im Zusammenhang mit StyleVision	92
5.8	Fenster "Übersicht"	93
5.9	Fenster "Meldungen"	94

6 Mapping zwischen Komponenten 96

6.1	Datenmappingmethoden (Standard / Mixed Content / Sub-Einträge kopieren)	99
6.1.1	Zielorientiert / Standard-Mapping	100
6.1.2	Quellorientiertes / Mixed Content Mapping	101
	<i>Mappen von Mixed Content</i>	101
	<i>Mixed Content-Beispiel</i>	106
	<i>Mappen von Mixed Content-Datenelementen mit Hilfe eines quellorientierten Mappings</i>	107
6.1.3	"Alles kopieren" Konnektoren	109
6.2	Verbindungseinstellungen	112
6.3	Verschieben von Konnektoren	114
6.4	Verbindungen und Mapping-Ergebnisse	117
6.5	Fehlende Datenelemente	118
6.6	Verkettete Mappings / Weiterleitungskomponenten	122
6.6.1	Verkettete Mappings - Weiterleitung aktiv	124
6.6.2	Verkettete Mappings - Weiterleitung deaktiviert	129
6.6.3	Beispiel für ein verkettetes Mapping	134
6.7	Arbeiten mit Funktionen	136
6.8	Vorschau auf das Transformationsergebnis	139
6.9	Validieren von Mappings und Mapping-Ausgabe	142
6.10	Schleifen, Gruppen und Hierarchien	145
6.11	Mapping-Regeln und -Strategien	146

7 Reihenfolge der Verarbeitung von Mapping-Komponenten 152

8	Built-In-Ausführungsprozessor	156
9	Globale Ressourcen	158
9.1	Globale Ressourcen - Dateien	159
9.1.1	Definieren / Hinzufügen einer globalen Ressource	160
9.1.2	Zuweisen einer globalen Ressource	163
9.1.3	Verwendung / Aktivierung einer globalen Komponente	165
9.2	Globale Ressourcen - Ordner	167
9.3	Globale Ressourcen - Arbeitsablauf in Applikationen	171
9.3.1	Starten des Applikationsarbeitsablaufs	175
9.4	Globale Ressourcen - Datenbanken	178
9.5	Globale Ressourcen - Eigenschaften	183
10	Dynamische Input/Output-Dateien pro Komponente	188
10.1	Dynamische Dateinamen - Input / Output	190
10.2	Dynamische Dateinamen als Input-Parameter	193
10.3	Mehrere XML-Dateien anhand einer einzigen XML-Quelldatei	195
10.4	Mehrere XML-Dateien pro Tabelle	198
10.5	Mehrere XML-Dateien anhand von Excel-Zeilen	200
10.6	Relative und absolute Dateipfade	205
11	Zwischenvariablen	210
11.1	Variablen - Anwendungsfälle	216
12	Digitale Signaturen	220
12.1	XML-Signatureinstellungen	225
13	MapForce Tipps...	230
13.1	Filtern - Abrufen dynamischer Daten, Lookup-Tabelle	232
13.2	Filterkomponenten - Tipps	234
13.3	Sortierkomponente - Sortieren von Input-Sequenzen	236
13.4	Wertezuordnung - Transformieren von Input-Daten	244

13.4.1	Übergeben von unveränderten Daten über eine Wertezuordnung	248
13.4.2	Eigenschaften der Wertezuordnungskomponente	251
13.5	Ersetzen von Sonderzeichen in Datenbankdaten	253
13.6	Aggregatfunktionen: min, max, sum, count, avg	255
13.7	Mappen mehrerer Tabellen auf eine XML-Datei	257
13.8	Mappings und das Root-Element von Zieldokumenten	259
13.9	Boolscher Vergleich von Input Nodes	260
13.10	Prioritätskontext Node/Datenelement	262
13.11	Zusammenführen mehrerer Dateien in einer Zieldatei	264
13.12	Befehlszeilenparameter	266
13.13	Befehlszeile - Definieren von Input-Parametern	273
13.14	Input-Parameter - Standardeinstellungen und Einstellungen für die Vorschau	275
13.15	Komponentennamen	278
13.16	Filtern von Datenbankdaten nach Datum	280
13.17	Node-Tests, Position und Gruppierung	281
13.17.1	Mappen fehlender Nodes - mit Hilfe von Not-exists	283
13.17.2	Position von Kontextelementen in einer Sequenz	285
13.17.3	Gruppieren von Nodes / Node-Inhalt	287
13.18	Verwendung von DTDs als "Schema-Komponenten"	294
13.19	Überprüfung der Typkonvertierung	295
13.20	Ausnahmeereignisse	296
13.21	Katalogdateien in MapForce	298
13.22	Derived XML-Schema-Typen - Mappen auf	302
13.23	Rekursives benutzerdefiniertes Mapping	304
13.23.1	Definieren eines rekursiven benutzerdefinierten Mappings	306

14 StyleVision Power Stylesheets in der Vorschau 314

14.1	Zuweisen einer SPS-Datei zu einer Komponente	316

15 Dokumentieren von Mapping-Projekten 320

15.1	Mitgelieferte SPS Stylesheets	326
15.2	Benutzerdefiniertes Design	329

16 Nullwerte / Nillable Werte 332

17 Datenbanken und MapForce 338

17.1	Installation von Datenbank-Clients / -Treibern	339
17.1.1	SQL Server	340
	<i>SQL Server 2000</i>	340
	<i>SQL Server 2005</i>	340
	<i>SQL Server 2008</i>	340
17.1.2	Oracle	341
	<i>Oracle 9i</i>	341
	<i>Oracle 10g</i>	341
	<i>Oracle 11g</i>	341
	<i>Installation eines Oracle Client</i>	341
17.1.3	IBM DB2	343
	<i>DB2 Version 8 / 9</i>	343
	<i>DB2 für i 5.4</i>	343
17.1.4	MySQL	344
	<i>MySQL 4 / 5</i>	344
17.1.5	PostgreSQL	345
	<i>PostgreSQL 8.x</i>	345
17.1.6	Sybase	346
	<i>Sybase 12</i>	346
17.1.7	Microsoft Access	347
17.1.8	Firebird	348
	<i>Firebird 2.0.5 / 2.1.2</i>	348
17.2	Tutorial: Mappen von XML-Daten auf Datenbanken	349
17.2.1	Einrichten von XML-auf-Datenbank Mappings	350
17.2.2	Einfügen von Datenbanken - Anpassen der Tabellenvorschau	354
17.2.3	Komponenten und Tabellenbeziehungen	357
17.2.4	Datenbankaktion: Einfügen	359
	<i>Einfügen von Daten in eine Datenbanktabelle</i>	359
	<i>Einfügen von Tabellen und damit in Zusammenhang</i> <i>stehenden untergeordneten Tabellen</i>	362
17.2.5	Datenbankaktion: Aktualisieren	367
	<i>Aktualisieren von Datenbankfeldern</i>	367
	<i>Aktualisieren und Hinzufügen von neuen Datensätzen</i>	370
	<i>Aktualisieren und Löschen von Child-Daten</i>	375
17.2.6	Datenbankaktion: Löschen	379
17.2.7	Datenbankaktion: Ignorieren	383
17.2.8	Generieren von Datenbankausgabewerten	386
17.3	Verwendung des Verbindungsassistenten	387

17.4	JDBC-Verbindungen	394
17.5	Tabellenaktionen, Schlüsseleinstellungen, Transaktionsverarbeitung	398
17.5.1	Tabellenaktionen	400
17.6	Datenbankfunktionsmatrix	404
17.6.1	Datenbank-Info - MS Access	405
17.6.2	Datenbank-Info - MS SQL Server	407
17.6.3	Datenbank-Info - Oracle	409
17.6.4	Datenbank-Info - MySQL	411
17.6.5	Datenbank-Info - Sybase	413
17.6.6	Datenbank-Info - IBM DB2	415
17.7	Datenbankbeziehungen - beibehalten / aufheben	417
17.8	Lokale Beziehungen - Erzeugen von Datenbankbeziehungen	420
17.9	Mappen großer Datenbanken mit Hilfe von MapForce	424
17.9.1	Import der gesamten Datenbank	425
17.9.2	Datenbank-Teilimport	426
17.10	Datenbank, Null-Verarbeitungsfunktionen	428
17.11	SQL WHERE/ORDER-Komponente	430
17.11.1	SQL WHERE/ORDER-Operatoren	434
17.12	Mappen von XML-Daten von / auf Datenbanken	438
17.13	IBM DB2 - Mappen von XML-Daten von / auf Datenbanken	442
17.13.1	Abfragen und Mappen von XML-Daten in IBM DB2	448
17.13.2	Mappen von XML-Daten - IBM DB2 als Ziel	450
17.13.3	Mappen von Daten - Datenbank auf Datenbank	452
17.14	SQL Server 2005 - Mappen von XML-Daten	455
17.15	Direktes Abfragen von Datenbanken - Register "Datenbankabfrage"	460
17.15.1	Auswählen / Herstellen einer DB-Verbindung	462
17.15.2	Auswählen einer Datenbank - Globale Ressourcen	466
17.15.3	Durchführen von Datenbankabfragen	469
17.15.4	Datenbankabfrage - SQL-Fenster	470
	Generieren von SQL-Anweisungen.....	471
	Ausführen von SQL-Anweisungen.....	472
	Speichern und Öffnen von SQL Scripts.....	473
	SQL Editor-Funktionen.....	473
	Autokomplettierung.....	474
	Auskommentieren von Text.....	475
	Verwenden von Lesezeichen.....	476
	Einfügen von Regionen.....	476
17.15.5	Datenbankabfrage - Browser-Fenster	478
	Filtern und Suchen von Datenbankobjekten.....	479
	Kontextoptionen in der Bowser-Ansicht.....	481
17.15.6	Datenbankabfrage - Ergebnis- und Meldungsregister	483

17.15.7 Datenbankabfrage - Einstellungen	486
<i>SQL Editor Optionen</i>	486
Generation.....	487
Autokomplettierung	488
Ergebnisansicht.....	489
Code-Generierung.....	489
17.16 SELECT-Anweisungen als virtuelle Tabellen	491
17.16.1 Erstellen von SELECT-Anweisungen	492
17.16.2 Beispiel für eine SELECT-Anweisung	495

18 Mappen von CSV und Textdateien 498

18.1 Mappen von CSV-Dateien auf XML	499
18.2 Mappen von XML auf CSV, Iterieren durch Datenelemente	503
18.3 Hierarchien für CSV-Dateien und Dateien mit fester Länge	506
18.4 CSV-Dateioptionen	510
18.5 Mappen von Textdateien mit fester Länge (auf eine Datenbank)	514
18.5.1 Optionen für Textdateien mit fester Länge	522
18.6 Mappen von Datenbanken auf CSV/Text-Dateien	526

19 MapForce FlexText 530

19.1 Übersicht	531
19.2 FlexText Tutorial	534
19.3 Erstellen von Trennbedingungen	537
19.4 Definieren mehrerer Bedingungen pro Container/Fragment	540
19.5 Verwendung von FlexText-Vorlagen in MapForce	545
19.5.1 FlexText Datenquellen	548
19.6 Verwenden von FlexText als Zielkomponente	549
19.7 FlexText Referenz	551
19.7.1 Mehrmals trennen	552
<i>Modus - Feste Länge</i>	553
<i>Modus - Getrennt (beliebig)</i>	553
<i>Modus - Getrennt (zeilenbasiert)</i>	555
<i>Modus - Getrennt (Beginnt mit...)</i>	556
19.7.2 Einmal trennen	558
<i>Modus - Feste Länge</i>	558
<i>Modus - Getrennt (beliebig)</i>	559
<i>Modus - Getrennt (zeilenbasiert)</i>	560
19.7.3 Aufteilen	562
19.7.4 Node	566

19.7.5	Ignorieren	568
19.7.6	Als CSV speichern (getrennt)	570
19.7.7	Als FLF speichern (feste Länge)	576
19.7.8	Als Wert speichern	579
20	EDI - UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc	582
20.1	EDI-Terminologie	584
20.2	Mappen von UN/EDIFACT auf ein XML-Schema	585
20.3	Validierung von EDI-Komponenten	594
20.4	X12 997 - Functional Acknowledgment	598
20.5	X12 999 - Implementation Acknowledgment	600
20.6	TA1-Segment	602
20.7	Mehrere EDI-Nachrichten pro Komponente	604
20.8	UN/EDIFACT und ANSI X12 als Zielkomponenten	607
20.8.1	Validierung des UN/EDIFACT-Zieldokuments	611
20.8.2	Validierung des ANSI X12-Zieldokuments	614
20.9	Anpassen einer EDIFACT Message	617
20.9.1	EDIFACT: Anpassung	618
20.9.2	Globale Anpassung	621
20.9.3	Inline-Anpassung	623
20.9.4	Mapping-Beispiel Orders	625
20.10	Anpassen einer ANSI X12-Transaktion	627
20.10.1	X12-Quelldateien und mehrere identische Felder	628
20.10.2	X12 Anpassung - Vorbereitung	630
20.10.3	Globale Anpassung	632
20.10.4	Inline-Anpassung	633
20.10.5	X12-Mapping-Beispiel	635
20.11	Aufgliedern zusammengeführter Einträge in separate Nodes/Datenelemente	636
20.12	Aktualisieren von config-Dateien zur Unterstützung mehrerer Nachrichten	639
20.13	SAP IDOCS	641
20.14	IATA PADIS	643
20.15	HIPAA X12	646
20.15.1	HIPAA Transactions	649
21	Mappen von HL7 v3.x von/auf XML-Schemas	654

22	MS OOXML Excel-Dateien (und höhere Versionen)	658
22.1	Definieren der mapbaren Datenelemente einer Excel Arbeitsmappe	659
22.2	Verwenden von Excel-Namen für Bereiche und Excel-Tabellen	665
22.3	Hinzufügen, Definieren und Verschieben von Zeilenbereichen	668
22.4	Zellenbereich auswählen - Optionen	674
22.5	Mappen von Excel-Dateien auf XML	678
22.6	Mappen von Datenbankdaten auf Excel	681
23	Bibliotheken und Funktionen	686
23.1	Definieren benutzerdefinierter Funktionen	687
23.1.1	Funktionsparameter	693
23.1.2	Inline-Funktionen und reguläre benutzerdefinierte Funktionen	697
23.1.3	Erstellen einer einfachen Lookup-Funktion	699
23.1.4	Komplexe benutzerdefinierte Funktion - XML-Node als Input	704
	<i>Definieren komplexer Input-Komponenten</i>	705
23.1.5	Komplexe benutzerdefinierte Funktion - XML Node als Output	710
	<i>Definieren komplexer Output-Komponenten</i>	710
23.1.6	Benutzerdefinierte Funktion - Beispiel	716
23.2	Hinzufügen benutzerdefinierter XSLT- und XQuery-Funktionen	721
23.2.1	Hinzufügen benutzerdefinierter Java .class- und .NET DLL-Funktionen ...	722
23.2.2	Benutzerdefinierte XSLT 1.0-Funktionen	724
23.2.3	Benutzerdefinierte XSLT 2.0-Funktionen	728
23.2.4	Benutzerdefinierte XQuery-Funktionen	729
23.2.5	Aggregatfunktionen - Aggregieren von Nodes in XSLT1 und 2	730
23.3	Hinzufügen benutzerdefinierter Java, C# und C++-Funktionsbibliotheken	733
23.3.1	Konfigurieren der MFF-Datei	735
23.3.2	Definieren der Komponenten-Benutzerschnittstelle	737
23.3.3	Einzelheiten zur Funktionsimplementierung	739
23.3.4	Schreiben eigener Bibliotheken	740
	<i>Erstellen einer Java-Bibliothek</i>	740
	<i>Erstellen einer C#-Bibliothek</i>	742
	<i>Erstellen einer C++-Bibliothek</i>	743
23.4	Java- und .NET-Funktionen - Details	747
23.5	Referenz Funktionen	750
23.5.1	core	752
	<i>aggregates (Aggregatfunktionen)</i>	752

	<i>conversion functions (Konvertierungsfunktionen)</i>	755
	<i>file path functions (Dateipfadfunktionen)</i>	764
	<i>generator functions (Generierungsfunktionen)</i>	766
	<i>logical functions (logische Funktionen)</i>	768
	<i>math functions (mathematische Funktionen)</i>	770
	<i>node functions (Node-Funktionen)</i>	772
	<i>sequence functions (Sequenzfunktionen)</i>	775
	<i>string functions (String-Funktionen)</i>	776
	Tokenize Beispiele.....	780
	Regular Expressions.....	784
23.5.2	edifact	788
23.5.3	lang	790
	<i>QName functions (QName-Funktionen)</i>	790
	<i>datetime functions (Datums- und Uhrzeitfunktionen)</i>	790
	<i>generator functions (Generierungsfunktionen)</i>	794
	<i>logical functions (logische Funktionen)</i>	794
	<i>math functions (mathematische Funktionen)</i>	794
	<i>string functions (String-Funktionen)</i>	796
23.5.4	xbrl	800
23.5.5	xlsx	801
23.5.6	xpath2	802
	<i>accessors (Accessor-Funktionen)</i>	802
	<i>anyURI functions (Alle URI-Funktionen)</i>	802
	<i>boolean functions (Boolesche Funktionen)</i>	803
	<i>constructors (Konstruktoren)</i>	803
	<i>context functions (Kontextfunktionen)</i>	804
	<i>durations, date and time functions (Zeitdauer-, Datums- und Uhrzeitfunktionen)</i>	805
	<i>node functions (Node-Funktionen)</i>	807
	<i>numeric functions (numerische Funktionen)</i>	808
	<i>qname-related functions (QName-Funktionen)</i>	809
	<i>string functions (String-Funktionen)</i>	809
23.5.7	xslt	812
	<i>xpath functions (XPath-Funktionen)</i>	812
	<i>xslt functions (XSLT-Funktionen)</i>	814

24 Implementieren von Webservices 818

24.1	WSDL - Info - unterstützte Protokolle	819
24.2	Erstellen von Webservice-Projekten anhand von WSDL-Dateien	822
24.3	Generieren von Java Webservices mit MapForce	825

24.3.1	Verwendung des Webservice - getPerson-Operation	827
24.3.2	Verwendung des Webservice - putPerson-Operation	829
24.4	Generieren von C# Webservices mit MapForce	831
24.5	Webservice Fehler	833
25	Aufrufen von Webservices	836
25.1	Aufruf der Webservice-Funktion getCityTime	837
25.2	Aufruf des Webservice getPerson	842
26	XBRL-Mapping	846
26.1	Einfügen von XBRL-Dokumenten	847
26.2	XBRL-Komponenten	851
26.2.1	XBRL-Hypercubes	859
26.2.2	XBRL-Wertezuordnungsfunktion	865
26.3	Beispiel Excel - XBRL	867
26.3.1	Beschreibung des Beispiels	868
26.4	Beispiel Datenbank - XBRL	871
27	Unterstützung für QName	874
28	MapForce Plug-in für MS Visual Studio	878
28.1	Öffnen von MapForce Dateien in Visual Studio	879
28.2	Unterschiede zwischen Visual Studio und Standalone-Versionen	881
29	MapForce Plug-in für Eclipse	884
29.1	Manuelle Installation des MapForce Plug-in	886
29.2	Manuelle Installation des MapForce Plug-in - Eclipse 3.6	891
29.3	Starten von Eclipse und Verwendung des MapForce Plug-in	897
29.4	MapForce / Editor, Ansicht und Perspektiven	900
29.5	Importieren des Ordners MapForce examples in den Navigator	902
29.6	Erstellen neuer MapForce Dateien (Mapping- und Projektdatei)	904
29.7	MapForce Codegenerierung	905
29.7.1	Manuelles Erstellen von Mapping-Code	906
29.7.2	Verwenden von MapForce Eclipse-Projekten zur automatischen Erstellung eines Build	
29.7.3	Hinzufügen der MapForce Nature zum vorhandenen Eclipse-Projekt	910

29.8	Erweitern des MapForce Plug-in	911

30 Benutzerreferenz 916

30.1	Datei	917
30.2	Bearbeiten	922
30.3	Einfügen	923
30.4	Projekt	927
30.5	Komponente	929
30.6	Verbindung	938
30.7	Funktion	943
30.8	Ausgabe	946
30.9	Ansicht	948
30.10	Extras	950
30.11	Fenster	958
30.12	Hilfe	959
30.12.1	Inhaltsverzeichnis, Index, Suchen	960
30.12.2	Aktivierung, Bestellformular, Registrieren, Updates	961
30.12.3	Weitere Befehle	963

31 Code Generator 966

31.1	Einführung in Code Generator	967
31.2	Was ist neu	969
31.3	Erzeugen von Programmcode	971
31.3.1	Generieren von Java-Code	973
31.3.2	Generieren von C#-Code	975
31.3.3	Generieren von C++-Code	978
31.4	Mapping-Beispiel mit Generierung von Code	981
31.5	Integrieren von MapForce Code in Ihre Applikation	984
31.5.1	MapForce Code in Java-Applikationen	985
31.5.2	MapForce Code in C#-Applikationen	987
31.5.3	MapForce Code in C++-Applikationen	989
31.5.4	Unterstützung von Datenstreams	991
31.6	Verwendung der generierten Codebibliothek	995
31.6.1	Beispielschema	997
31.6.2	Verwendung der generierten Java-Bibliothek	998
31.6.3	Verwendung der generierten C++-Bibliothek	1005
31.6.4	Verwendung der generierten C#-Bibliothek	1013

31.6.5	Verwendung von generiertem Code, der mit älteren Versionen kompatibel ist)	
	<i>Erzeugen von XML-Dateien (XMLSpy 2006 - 2007)</i>	1020
	<i>Erzeugen von XML-Dateien (XMLSpy 2005)</i>	1022
	<i>Öffnen und Parsen vorhandener XML-Dateien (XMLSpy 2006 - 2007)</i>	1024
	<i>Öffnen und Parsen vorhandener XML-Dateien (XMLSpy 2005)</i>	1027
31.7	Codegenerierung Tipps	1030
31.8	Code Generator-Optionen	1032
31.9	Der Weg zu SPL (Spy Programming Language)	1034
31.9.1	Grundlegende SPL-Struktur	1035
31.9.2	Deklarationen	1036
31.9.3	Variablen	1038
31.9.4	Vordefinierte Variablen	1039
31.9.5	Erstellen von Ausgabedateien	1040
31.9.6	Operatoren	1041
31.9.7	Bedingungen	1042
31.9.8	Collections und foreach	1043
31.9.9	Subroutinen	1045
	<i>Deklaration einer Subroutine</i>	1045
	<i>Subroutinenaufruf</i>	1046
	<i>Beispiel für eine Subroutine</i>	1047
31.9.10	Integrierte Typen	1049
	<i>Bibliothek</i>	1049
	<i>Namespace</i>	1049
	<i>Typ</i>	1049
	<i>Member</i>	1050
	<i>NativeBinding</i>	1051
	<i>Facets</i>	1051
	<i>Altes Objektmodell (bis v2007)</i>	1052
	<i>Namespace</i>	1052
	<i>Klasse</i>	1052
	<i>Member</i>	1053
	<i>Facet</i>	1055
	<i>Enumeration</i>	1056
	<i>Pattern</i>	1056

32 Die MapForce API 1058

32.1	Übersicht	1059
32.1.1	Objektmodell	1060
32.1.2	Beispiel: Codegenerierung	1061

32.1.3	Beispiel: Mapping-Ausführung	1063
32.1.4	Beispiel: Projektunterstützung	1067
32.1.5	Behandlung von Fehlern	1071
32.1.6	Programmiersprachen	1073
	<i>C#</i>	1073
	<i>Java</i>	1078
	Java-Beispielprojekt.....	1079
	<i>JScript</i>	1082
	Applikation starten.....	1083
	Einfacher Dokumentaufruf.....	1084
	Code generieren.....	1085
32.2	Objektreferenz	1088
32.2.1	Applikation	1089
	<i>Events</i>	1090
	OnDocumentOpened.....	1090
	OnProjectOpened.....	1090
	OnShutdown.....	1090
	<i>ActiveDocument</i>	1090
	<i>ActiveProject</i>	1091
	<i>Application</i>	1091
	<i>Documents</i>	1091
	<i>Edition</i>	1091
	<i>GlobalResourceConfig</i>	1091
	<i>GlobalResourceFile</i>	1092
	<i>HighlightSerializedMarker</i>	1092
	<i>IsAPISupported</i>	1092
	<i>MajorVersion</i>	1092
	<i>MinorVersion</i>	1093
	<i>Name</i>	1093
	<i>NewDocument</i>	1093
	<i>NewProject</i>	1093
	<i>OpenDocument</i>	1094
	<i>OpenProject</i>	1094
	<i>OpenURL</i>	1094
	<i>Options</i>	1094
	<i>Parent</i>	1094
	<i>Quit</i>	1095
	<i>ServicePackVersion</i>	1095
	<i>Status</i>	1095
	<i>Visible</i>	1095
	<i>WindowHandle</i>	1096
32.2.2	AppOutputLine	1097

	<i>Application</i>	1097
	<i>ChildLines</i>	1097
	<i>GetCellCountInLine</i>	1098
	<i>GetCellIcon</i>	1098
	<i>GetCellSymbol</i>	1098
	<i>GetCellText</i>	1098
	<i>GetCellTextDecoration</i>	1099
	<i>GetIsCellText</i>	1099
	<i>GetLineCount</i>	1099
	<i>GetLineSeverity</i>	1099
	<i>GetLineSymbol</i>	1099
	<i>GetLineText</i>	1100
	<i>GetLineTextEx</i>	1100
	<i>GetLineTextWithChildren</i>	1100
	<i>GetLineTextWithChildrenEx</i>	1100
	<i>Parent</i>	1101
32.2.3	<i>AppOutputLines</i>	1102
	<i>Application</i>	1102
	<i>Count</i>	1102
	<i>Item</i>	1102
	<i>Parent</i>	1102
32.2.4	<i>AppOutputLineSymbol</i>	1104
	<i>Application</i>	1104
	<i>GetSymbolHREF</i>	1104
	<i>GetSymbolID</i>	1104
	<i>IsSymbolHREF</i>	1104
	<i>Parent</i>	1105
32.2.5	<i>Component</i>	1106
	<i>Application</i>	1106
	<i>CanChangeInputInstanceFile</i>	1106
	<i>CanChangeOutputInstanceFile</i>	1107
	<i>GenerateOutput</i>	1107
	<i>GetRootDatapoint</i>	1107
	<i>HasIncomingConnections</i>	1108
	<i>HasOutgoingConnections</i>	1108
	<i>ID</i>	1108
	<i>InputInstanceFile</i>	1109
	<i>IsParameterInputRequired</i>	1109
	<i>IsParameterSequence</i>	1109
	<i>Name</i>	1109
	<i>OutputInstanceFile</i>	1110

	<i>Parent</i>	1110
	<i>Preview</i>	1110
	<i>Schema</i>	1111
	<i>SubType</i>	1111
	<i>Type</i>	1111
	<i>UsageKind</i>	1111
32.2.6	<i>Components</i>	1112
	<i>Application</i>	1112
	<i>Count</i>	1112
	<i>Item</i>	1112
	<i>Parent</i>	1112
32.2.7	<i>Connection</i>	1114
	<i>Application</i>	1114
	<i>ConnectionType</i>	1114
	<i>Parent</i>	1114
32.2.8	<i>Datapoint</i>	1115
	<i>Application</i>	1115
	<i>GetChild</i>	1115
	<i>Parent</i>	1115
32.2.9	<i>Document</i>	1116
	<i>Events</i>	1117
	<i>OnDocumentClosed</i>	1117
	<i>OnModifiedFlagChanged</i>	1117
	<i>Activate</i>	1117
	<i>Application</i>	1117
	<i>Close</i>	1117
	<i>CreateUserDefinedFunction</i>	1118
	<i>FindComponentByID</i>	1118
	<i>FullName</i>	1118
	<i>GenerateCHashCode</i>	1118
	<i>GenerateCodeEx</i>	1119
	<i>GenerateCppCode</i>	1119
	<i>GenerateJavaCode</i>	1119
	<i>GenerateOutput</i>	1120
	<i>GenerateOutputEx</i>	1120
	<i>GenerateXQuery</i>	1120
	<i>GenerateXSLT</i>	1120
	<i>GenerateXSLT2</i>	1121
	<i>HighlightSerializedMarker</i>	1121
	<i>JavaSettings_BasePackageName</i>	1121
	<i>MainMapping</i>	1122

	<i>MapForceView</i>	1122
	<i>Mappings</i>	1122
	<i>Name</i>	1122
	<i>OutputSettings_ApplicationName</i>	1122
	<i>OutputSettings_Encoding (nicht mehr verwendet)</i>	1123
	<i>Parent</i>	1123
	<i>Path</i>	1123
	<i>Save</i>	1123
	<i>SaveAs</i>	1124
	<i>Saved</i>	1124
32.2.10	<i>Documents</i>	1125
	<i>ActiveDocument</i>	1125
	<i>Application</i>	1125
	<i>Count</i>	1125
	<i>Item</i>	1126
	<i>NewDocument</i>	1126
	<i>OpenDocument</i>	1126
	<i>Parent</i>	1126
32.2.11	<i>ErrorMarker</i>	1127
	<i>Application</i>	1127
	<i>DocumentFileName</i>	1127
	<i>ErrorLevel</i>	1127
	<i>Highlight</i>	1127
	<i>Serialization</i>	1128
	<i>Text</i>	1128
	<i>Parent</i>	1128
32.2.12	<i>ErrorMarkers</i>	1129
	<i>Application</i>	1129
	<i>Count</i>	1129
	<i>Item</i>	1129
	<i>Parent</i>	1129
32.2.13	<i>MapForceView</i>	1131
	<i>Active</i>	1131
	<i>ActiveMapping</i>	1131
	<i>ActiveMappingName</i>	1131
	<i>Application</i>	1132
	<i>HighlightMyConnections</i>	1132
	<i>HighlightMyConnectionsRecursive</i>	1132
	<i>InsertWSDLCall</i>	1132
	<i>InsertXMLFile (nicht mehr verwendet)</i>	1133
	<i>InsertXMLSchema (nicht mehr verwendet)</i>	1133

	<i>InsertXMLSchemaWithSample (nicht mehr verwendet)</i>	1133
	<i>Parent</i>	1134
	<i>ShowItemTypes</i>	1134
	<i>ShowLibraryInFunctionHeader</i>	1134
32.2.14	Mapping	1136
	<i>Application</i>	1136
	<i>Components</i>	1136
	<i>CreateConnection</i>	1136
	<i>InsertFunctionCall</i>	1137
	<i>InsertXMLFile</i>	1137
	<i>InsertXMLSchema</i>	1138
	<i>InsertXMLSchemaInputParameter</i>	1138
	<i>InsertXMLSchemaOutputParameter</i>	1139
	<i>IsMainMapping</i>	1139
	<i>Name</i>	1139
	<i>Parent</i>	1140
32.2.15	Mappings	1141
	<i>Application</i>	1141
	<i>Count</i>	1141
	<i>Item</i>	1141
	<i>Parent</i>	1141
32.2.16	Optionen	1143
	<i>Application</i>	1143
	<i>CodeDefaultOutputDirectory</i>	1143
	<i>CompatibilityMode</i>	1144
	<i>CppSettings_DOMType</i>	1144
	<i>CppSettings_GenerateVC6ProjectFile</i>	1144
	<i>CppSettings_GenerateVSProjectFile</i>	1144
	<i>CppSettings_LibraryType</i>	1145
	<i>CppSettings_UseMFC</i>	1145
	<i>CSharpSettings_ProjectType</i>	1145
	<i>DefaultOutputByteOrder</i>	1146
	<i>DefaultOutputByteOrderMark</i>	1146
	<i>DefaultOutputEncoding</i>	1146
	<i>GenerateWrapperClasses</i>	1146
	<i>JavaSettings_ApacheAxisVersion (nicht mehr verwendet)</i>	1147
	<i>Parent</i>	1147
	<i>ShowLogoOnPrint</i>	1147
	<i>ShowLogoOnStartup</i>	1147
	<i>UseGradientBackground</i>	1148
	<i>WrapperClassesVersion</i>	1148

	<i>XSLTDefaultOutputDirectory</i>	1148
32.2.17	Projekt (Enterprise oder Professional Edition)	1149
	<i>Events</i>	1149
	<i>OnProjectClosed</i>	1149
	<i>_NewEnum</i>	1150
	<i>AddActiveFile</i>	1150
	<i>AddFile</i>	1151
	<i>Application</i>	1151
	<i>Close</i>	1151
	<i>Count</i>	1151
	<i>CreateFolder</i>	1152
	<i>FullName</i>	1152
	<i>GenerateCode</i>	1152
	<i>GenerateCodeEx</i>	1152
	<i>GenerateCodeIn</i>	1153
	<i>GenerateCodeInEx</i>	1153
	<i>InsertWebService</i>	1153
	<i>Item</i>	1154
	<i>Java_BasePackageName</i>	1154
	<i>Name</i>	1154
	<i>Output_Folder</i>	1154
	<i>Output_Language</i>	1155
	<i>Output_TextEncoding</i>	1155
	<i>Parent</i>	1155
	<i>Path</i>	1155
	<i>Save</i>	1156
	<i>Saved</i>	1156
32.2.18	ProjectItem (Enterprise oder Professional Edition)	1157
	<i>_NewEnum</i>	1157
	<i>AddActiveFile</i>	1158
	<i>AddFile</i>	1158
	<i>Application</i>	1158
	<i>CodeGenSettings_Language</i>	1158
	<i>CodeGenSettings_OutputFolder</i>	1159
	<i>CodeGenSettings_UseDefault</i>	1159
	<i>Count</i>	1159
	<i>CreateFolder</i>	1159
	<i>CreateMappingForProject</i>	1160
	<i>GenerateCode</i>	1160
	<i>GenerateCodeEx</i>	1160
	<i>GenerateCodeIn</i>	1161

	<i>GenerateCodeInEx</i>	1161
	<i>Item</i>	1161
	<i>Kind</i>	1162
	<i>Name</i>	1162
	<i>Open</i>	1162
	<i>Parent</i>	1162
	<i>QualifiedName</i>	1163
	<i>Remove</i>	1163
	<i>WSDLFile</i>	1163
32.3	Enumerationen	1164
32.3.1	ENUMApacheAxisVersion (nicht mehr verwendet)	1165
32.3.2	ENUMApplicationStatus	1166
32.3.3	ENUMAppOutputLine_Severity	1167
32.3.4	ENUMAppOutputLine_TextDecoration	1168
32.3.5	ENUMCodeGenErrorLevel	1169
32.3.6	ENUMComponentDatapointSide	1170
32.3.7	ENUMComponentSubType	1171
32.3.8	ENUMComponentType	1172
32.3.9	ENUMComponentUsageKind	1173
32.3.10	ENUMConnectionType	1174
32.3.11	ENUMDOMType	1175
32.3.12	ENUMLibType	1176
32.3.13	ENUMProgrammingLanguage	1177
32.3.14	ENUMProjectItemType	1178
32.3.15	ENUMProjectType	1179
32.3.16	ENUMSearchDatapointFlags	1180
32.3.17	ENUMViewMode	1181
32.3.18	ENUMWrapperClassesVersion	1182

33 ActiveX Integration 1184

33.1	Integration auf Applikationsebene	1185
33.1.1	Beispiel: HTML	1186
	<i>Instantiieren des Control</i>	1186
	<i>Hinzufügen einer Schaltfläche zum Öffnen eines Standarddokuments</i>	1186
	<i>Hinzufügen von Schaltflächen für die Codegenerierung</i>	1186
	<i>Verbindung zu benutzerdefinierten Events herstellen</i>	1187
33.2	Integration auf Dokumentebene	1189
33.2.1	Verwendung von MapForceControl	1190
33.2.2	Verwendung von MapForceControlDocument	1191

33.2.3	Verwendung von MapForceControlPlaceHolder	1192
33.2.4	Abfragen von MapForce Befehlen	1193
33.2.5	Beispiele	1194
	<i>C#</i>	1194
	Einführung	1194
	Platzieren des MapForceControl	1194
	Hinzufügen des Placeholder Control	1195
	Abrufen von Befehlsinformationen	1197
	Behandlung von Events	1199
	Testen des Beispiels	1200
	<i>HTML</i>	1202
	Instantiieren des MapForceControl	1202
	Erstellen eines Bearbeitungsfensters	1202
	Erstellen eines Projektfensters	1202
	Erstellen eines Placeholder für Eingabehilfenfenster	1203
	Erstellen einer benutzerdefinierten Symbolleiste	1203
	Erstellen weiterer Schaltflächen	1204
	Erstellen eines Event Handler zum Aktualisieren des Schaltflächenstatus	1205
	<i>Visual Basic</i>	1206
33.3	Befehlstabelle für MapForce	1207
33.3.1	Menü "Datei"	1208
33.3.2	Menü "Bearbeiten"	1209
33.3.3	Menü "Einfügen"	1210
33.3.4	Menü "Projekt"	1211
33.3.5	Menü "Komponente"	1212
33.3.6	Menü "Verbindung"	1213
33.3.7	Menü "Funktion"	1214
33.3.8	Menü "Ausgabe"	1215
33.3.9	Menü "Ansicht"	1216
33.3.10	Menü "Extras"	1217
33.3.11	Menü "Fenster"	1218
33.3.12	Menü "Hilfe"	1219
33.3.13	Befehle, die sich nicht im Hauptmenü befinden	1220
33.4	Aufrufen von MapForceAPI	1221
33.5	Objektreferenz	1222
33.5.1	MapForceCommand	1223
	<i>Accelerator</i>	1223
	<i>ID</i>	1223
	<i>IsSeparator</i>	1223
	<i>Label</i>	1223
	<i>StatusText</i>	1224
	<i>SubCommands</i>	1224
	<i>ToolTip</i>	1224

33.5.2	MapForceCommands	1225
	<i>Count</i>	1225
	<i>Item</i>	1225
33.5.3	MapForceControl	1226
	<i>Eigenschaften</i>	1226
	Appearance.....	1226
	Application.....	1227
	BorderStyle.....	1227
	CommandsList.....	1227
	EnableUserPrompts.....	1227
	IntegrationLevel.....	1227
	MainMenu.....	1228
	Toolbars.....	1228
	<i>Methoden</i>	1228
	Exec.....	1228
	Open.....	1228
	QueryStatus.....	1229
	<i>Events</i>	1229
	OnCloseEditingWindow.....	1229
	OnContextChanged.....	1229
	OnDocumentOpened.....	1229
	OnFileChangedAlert.....	1230
	OnLicenseProblem.....	1230
	OnOpenedOrFocused.....	1230
	OnToolWindowUpdated.....	1230
	OnUpdateCmdUI.....	1231
	OnValidationWindowUpdated.....	1231
33.5.4	MapForceControlDocument	1232
	<i>Eigenschaften</i>	1232
	Appearance.....	1232
	BorderStyle.....	1233
	Document.....	1233
	IsModified.....	1233
	Path.....	1233
	ReadOnly.....	1233
	<i>Methoden</i>	1234
	Exec.....	1234
	New.....	1234
	Open.....	1234
	QueryStatus.....	1234
	Reload.....	1235
	Save.....	1235
	SaveAs.....	1235
	<i>Events</i>	1235
	OnActivate.....	1235
	OnContextChanged.....	1236
	OnDocumentClosed.....	1236
	OnDocumentOpened.....	1236
	OnDocumentSaveAs.....	1236
	OnFileChangedAlert.....	1236

	OnModifiedFlagChanged.....	1236
	OnSetEditorTitle.....	1237
33.5.5	MapForceControlPlaceHolder	1238
	<i>Eigenschaften</i>	1238
	Label.....	1238
	PlaceholderWindowID.....	1238
	Project.....	1238
	<i>Methoden</i>	1239
	OpenProject	1239
	CloseProject.....	1239
	<i>Events</i>	1239
	OnModifiedFlagChanged.....	1239
	OnSetLabel.....	1240
33.5.6	Enumerationen	1241
	<i>ICActiveXIntegrationLevel</i>	1241
	<i>MapForceControlPlaceholderWindow</i>	1241

34 Anhänge 1244

34.1	Informationen zu den Prozessoren	1245
34.1.1	XSLT 1.0-Prozessor: Implementierungsinformationen	1246
34.1.2	XSLT 2.0-Prozessor: Implementierungsinformationen	1248
	<i>Allgemeine Informationen</i>	1248
	<i>XSLT 2.0-Elemente und -Funktionen</i>	1250
34.1.3	XQuery 1.0-Prozessor: Implementierungsinformationen	1251
34.1.4	XPath 2.0- und XQuery 1.0-Funktionen	1254
	<i>Allgemeine Informationen</i>	1254
	<i>Unterstützung von Funktionen</i>	1255
34.1.5	Erweiterungen	1259
	<i>Java-Erweiterungsfunktionen</i>	1259
	Benutzerdefinierte Klassendateien.....	1261
	Benutzerdefinierte Jar-Dateien	1263
	Java: Konstruktoren.....	1264
	Java: Statische Methoden und statische Felder.....	1264
	Java: Instanzmethoden und Instanzfelder.....	1265
	Datentypen: XPath/XQuery in Java.....	1266
	Datentypen: Java in XPath/XQuery.....	1267
	<i>.NET-Erweiterungsfunktionen</i>	1267
	.NET: Konstruktoren.....	1269
	.NET: Statische Methoden und statische Felder.....	1270
	.NET: Instanzmethoden und Instanzfelder.....	1271
	Datentypen: XPath/XQuery in .NET	1272
	Datentypen: .NET in XPath/XQuery.....	1273
	<i>MSXSL Scripts für XSLT</i>	1273
	<i>Altova Erweiterungsfunktionen</i>	1275
	Allgemeine Funktionen.....	1276

	Diagrammfunktionen	1279
	Diagrammdaten-XML-Struktur	1282
	Beispiel: Diagrammfunktionen	1287
34.2	Technische Daten	1291
34.2.1	OS- und Arbeitsspeichieranforderungen	1292
34.2.2	Altova XML Parser	1293
34.2.3	Altova XSLT- und XQuery-Prozessor	1294
34.2.4	Unicode-Unterstützung	1295
	<i>Windows XP</i>	1295
	<i>Rechts nach Links-Schreibrichtung</i>	1296
34.2.5	Internet-Verwendung	1297
34.3	Lizenzinformationen	1298
34.3.1	Electronic Software Distribution	1299
34.3.2	Software-Aktivierung und Lizenzüberwachung	1300
34.3.3	Rechte am geistigen Eigentum	1301
34.3.4	Altova Endbenutzer-Lizenzvereinbarung	1302

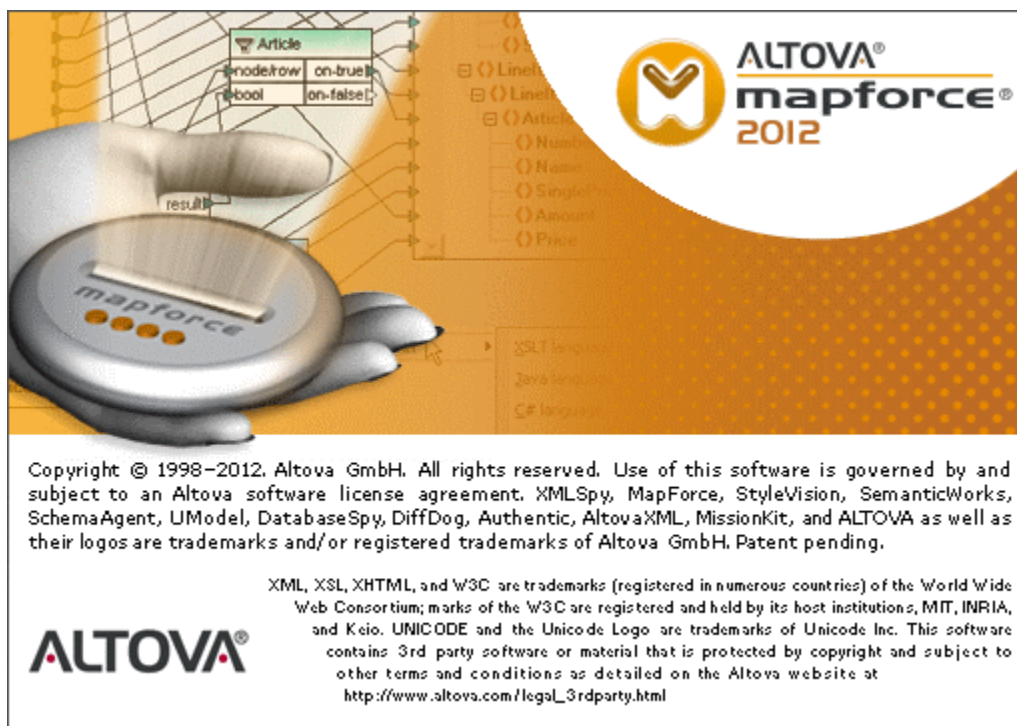
Index

Kapitel 1

MapForce 2012

1 MapForce 2012

MapForce® 2012 Enterprise Edition ist ein visuelles Mapping Tool für professionelle Datenintegrationsprojekte. MapForce® ist eine 32/64-Bit Windows Applikation, die auf Windows 7, Windows Vista, Windows Server 2003/2008 und Windows XP läuft. 64-Bit-Unterstützung steht für die Enterprise und die Professional Edition zur Verfügung.



Kapitel 2

Was ist neu...

2 Was ist neu...

MapForce Version **2012 R2** enthält die folgenden neuen Funktionen:

- Neue [Sortierkomponente](#) für den XSLT 2.0-, XQuery- und den Built-in-Ausführungsprozessor
- Benutzerdefinierte [Komponentennamen](#)
- Erweiterte SQL-Where-Funktionalität: [ORDER BY](#)

MapForce Version **2012** enthält die folgenden neuen Funktionen:

- [Streaming von Input/des Lesens](#) bei XML-, CSV- und FLF-Dateien - Built-in-Ausführungsprozessor
- Neuer Datenbankprozessor unterstützt direkte ODBC- und [JDBC-Verbindungen](#)
- [Automatisches Ausrichten](#) von Komponenten im Mapping-Fenster
- Neue Funktionen: [parse-date](#) und [parse-time](#)
- [Suchfunktion](#) auf dem Register/im Fenster "Projekt"
- Aufforderung eine Verbindung mit dem [Parent-Ziel-Node](#) herzustellen
- Spezielle Regeln, mit denen festgelegt wird, in welcher [Reihenfolge](#) Komponenten in einem Mapping verarbeitet werden
- Neuer Abschnitt in der MapForceAPI mit Codebeispielen in [Programmiersprachen](#)

MapForce Version 2011R3 enthält die folgenden neuen Funktionen:

- [Zwischenvariablen](#)
- Unterstützung für [mehrere Zeilenbereiche](#) in Excel-Komponenten
- Unterstützung für EDIX12 - [HIPAA](#) Implementation Guides
- Generierung der [X12 EDI 999](#) Implementation Acknowledgement-Komponente
- Digitale [Signaturen](#)
- Unterstützung für [US-GAAP 2011](#)
- Unterstützung für [.NET Framework 4.0](#) Assembly-Dateien
- Funktion zur Ausgabe von mit [StyleVision](#) formatierten Dokumenten über die Befehlszeile

MapForce Version 2011R2 enthält die folgenden neuen Funktionen:

- [Der Built-In-Ausführungsprozessor unterstützt nun die Streaming-Ausgabe](#)
- [Suchen von Funktionen](#) im Fenster "Bibliothek"
- [Reverse](#) mapping
- Erweiterbare [IF-ELSE](#)-Funktion
- Die Funktionen [Node Name](#) und [Parser](#)-Funktionen in der Core-Bibliothek
- Neues EDI-Format [IATA / PADIS](#)
- Möglichkeit [mehrere EDI-Nachrichten](#) pro Komponente zu verarbeiten
- Verbessertes [Dialogfeld "Datenbankaktionen"](#) mit Einstellungen zur integrierten Schlüsselgenerierung
- Neue Option zur Verwendung von StyleVision Power Stylesheets zur [Dokumentation](#) eines Mappings

MapForce Version 2011 enthält die folgenden neuen Funktionen:

- Möglichkeit, mit Hilfe von [StyleVision](#) Power Stylesheets, die StyleVision-Diagramme enthalten, eine Vorschau von Zielkomponenten zu erstellen
- Funktion zur Anzeige einer Vorschau von Zwischenkomponenten in einer [Mapping-Kette](#) von zwei oder mehr Komponenten, die mit einer Zielkomponente verbunden sind

(Weiterleitungsvorschau).

- Formatierungsfunktionen für [dateTime](#)- und [numbers](#)-Datentypen für alle unterstützten Programmiersprachen
- Verbesserungen an der [auto-number](#) Funktion
- Neue Zeitzonefunktionen: [remove-timezone](#) und [convert-to-utc](#)

MapForce Version 2010 Release 3 enthält die folgenden neuen Funktionen:

- Unterstützung für die Generierung von [Visual Studio 2010](#) Projektdateien für C# und C++
- Definieren einer Arbeitsblattzeile in einer Excel-Komponente als [Spaltennamen](#)
- Unterstützung für MSXML 6.0 bei der Generierung von C++-Code
- Unterstützung für [Nillable Werte](#) und das xsi:nil Attribut in XML-Instanzdateien
- Möglichkeit, die automatische [Konvertierung in Zieltypen](#) in XML-Dokumenten zu deaktivieren
- Unterstützung für [SAP IDocs](#) (Nur Enterprise Edition)

MapForce Version 2010 Release 2 enthält die folgenden neuen Funktionen:

- 64-Bit MapForce Enterprise / Professional Edition auf 64-Bit-Betriebssystemen: Windows Server 2003/2008, Windows XP, Windows Vista und Windows 7
- Unterstützung für [Excel 2010](#)
- Automatische Verbindung identischer [Child-Konnektoren](#) beim Verschieben eines Parent-Konnektors
- Unterstützung für Felder in der [SQL Where](#)-Komponente
- Möglichkeit [kompilierte Java](#) .class und .NET Assembly-Dateien hinzuzufügen
- Möglichkeit, Input-Strings für die weitere Verarbeitung [in Teile aufzuteilen](#)
- Validierung von UN/EDIFACT- und ANSI X12 EDI-[Quelldateien](#)
- Generierung von [X12 EDI 997](#)-Functional Acknowledgement-Komponenten

MapForce Version 2010 enthält die folgenden neuen Funktionen:

- [Mehrere Input/Output-Dateien](#) pro Komponente
- Verbesserte Unterstützung von [relativen Pfaden](#)
- Verwendung von [derived types](#) dank Unterstützung von xsi:type
- Neues internes Datentypsysteem
- Verbesserte [Navigation in benutzerdefinierten Funktionen](#)
- [Validierung](#) der EDI-Ausgabe in generiertem Code
- Unterstützung der EDIFACT Service Messages CONTRL, AUTACK und KEYMAN
- Unterstützung für mit WSDL 2.0 definierte Webservices
- Verbesserte Behandlung von [Mixed Content](#) in XML-Elementen

MapForce Version 2009 SP1 enthält die folgenden neuen Funktionen:

- [Die Reihenfolge der Parameter](#) in benutzerdefinierten Funktionen kann durch den Benutzer festgelegt werden
- Möglichkeit XML-Dateien, die entsprechend dem XML-Schema [nicht gültig sind](#), trotzdem zu validieren
- [Reguläre](#) benutzerdefinierte (Standard)funktionen unterstützen nun komplexe

hierarchische Parameter

- Xerces 3.x Unterstützung beim Generieren von C++ Code

MapForce Version 2009 enthält die folgenden neuen Funktionen:

- Unterstützung für [XBRL](#) und XBRL Dimension-Instanzdateien sowie für XBRL-Taxonomien als Quell- und Zielkomponenten
- [EDI HL7](#) Versionen 2.2 bis 2.6 als Quell- und Zielkomponenten
- [EDI HL7 Versionen 3.x](#) XML als Quell- und Zielkomponenten
- [Dokumentation](#) von Mapping-Projekten
- Native Unterstützung für XML-Felder in [SQL Server](#)
- [Gruppierung von Nodes](#) oder Node-Inhalt
- Möglichkeit zum Filtern von Daten auf Basis einer [Node-Position](#) in einer Sequenz
- [QName](#)-Unterstützung
- [Suchen](#) von Datenelementen/Nodes in Komponenten

MapForce Version 2008 Release 2 enthält die folgenden neuen Funktionen:

- Unterstützung für [Office Open XML Excel 2007 \(und höher\) \(*.xlsx\)](#) als Quell- und Zielkomponenten
- Unterstützung für [Streams](#) als Input/Output in Java- und C#-generiertem Code.
- Generierung von Visual Studio 2008-Projektdateien für C++ und C#
- Funktion zum automatischen [Generieren von XML-Schemas](#) für XML-Dateien.
- Unterstützung für [SOAP](#) Version 1.2.
- Neue Option [Beginnt mit...](#) für mehrmaliges Trennen in FlexText.
- Funktion zum [Entfernen von Datenbankschmanamen](#) aus dem generierten Code.
- [SQL-SELECT-Anweisungen](#) als virtuelle Tabellen in Datenbankkomponenten
- [Lokale Beziehungen](#) - On-the-fly-Erstellung von Primär-/Sekundärschlüsselbeziehungen
- Unterstützung für [globale Altova-Ressourcen](#).
- Verbesserte Leistung

MapForce Version 2008 enthält die folgenden neuen Funktionen:

- [Aggregatfunktionen](#)
- Lookup-Komponente für [Vertezuordnung](#)
- Optionen zur verbesserten XML-Ausgabe: [Pretty Print](#) für XML-Ausgabe, [XML-Schema](#) Referenz auslassen und [Kodierungseinstellungen](#) für einzelne Komponenten.
- diverse interne Updates

MapForce Version 2007 Release 3 enthält die folgenden neuen Funktionen:

- XML-Datenmapping von / auf Datenbanken - [IBM DB2](#) und [andere](#)
- [Direkte Abfragen](#) an Datenbanken
- [SQL-WHERE](#)-Filter und Assistent für SQL-Anweisungen
- Vollständige Unterstützung für alle [EDI X12](#) Releases von 3040 bis 5030
- Vollständige Unterstützung für [UN/EDIFACT](#) Messages der Verzeichnisse 93A bis 06B
- Optimierter [Code Generator](#) und verbesserte Dokumentation

Kapitel 3

Übersicht über MapForce

3 Übersicht über MapForce

Altova Website:  [Einführung in MapForce](#)

Was ist Mapping?

Im Prinzip wird der Inhalt einer Komponente auf eine andere Komponente gemappt, also abgebildet, bzw. dorthin transformiert. Ein XML- oder Textdokument, eine Datenbank, eine Excel 2007 (und höher), eine EDI- oder eine XBRL-Datei kann auf ein anderes XML-Zieldokument, ein CSV- /Textdokument, eine Excel 2007 (und höher)-Datei, eine EDI-Datei oder ein XBRL-Dokument oder eine Datenbank gemappt werden. Die Transformation erfolgt über ein automatisch generiertes XSLT 1.0 oder 2.0 Stylesheet, den Built-In-Ausführungsprozessor oder Programmcode, der generiert wird.

Außerdem kann MapForce in einer einzigen Komponente mehrere Input-Dateien eines Verzeichnisses verarbeiten und mehrere Dateien ebenfalls in einer einzigen Komponente ausgeben.

Bei der Erzeugung einer XSLT-Transformation wird ein **Quellschema** auf ein **Zielschema** gemappt. Dabei werden Elemente/Attribute im Quellschema mit anderen Elementen/Attributen im Zielschema "verbunden". Da ein XML-Dokument mit einer Schemadatei verknüpft ist und dadurch definiert wird, werden im Endeffekt zwei XML-Dokumente aufeinander gemappt.

MapForce® unterstützt:

- Grafisches Mapping auf und von jeder Kombination und Anzahl von:
 - XML-Schemas als Quell- oder Zieldatei
- Die **Professional** Edition unterstützt außerdem:
 - Flat Files: kommagetrennt (CSV) und Formate mit fester Länge als Quelle und Ziel
 - Relationale Datenbanken als Quelle und Ziel
- Die **Enterprise** Edition unterstützt außerdem:
 - EDI-Dateien: UN/EDIFACT, ANSI X12 einschließlich HIPAA, HL7 2.x, IATA PADIS und SAP IDocs als Quelle und Ziel
 - FlexText™-Dateien als Quelle und Ziel
 - Office Open XML Excel 2007 (und höher)-Dateien als Quelle und Ziel
 - XBRL-Instanzdateien und Taxonomien
- Automatische Codegenerierung
 - XSLT 1.0 und 2.0
- Die **Professional** Edition und **Enterprise** Edition unterstützen außerdem:
 - XQuery
 - Java, C# und C++
 - 64-Bit-Version
- On-the-fly-Transformation und Vorschau auf alle Mappings ohne Code-Generierung oder -Kompilierung
- die Anzeige einer Vorschau auf Zwischenkomponenten in einer Mapping-Kette bestehend aus zwei oder mehr mit einer Zielkomponente verbundenen Komponenten (Weiterleitungsvorschau)
- die Anzeige einer Vorschau auf Zielkomponenten mit Hilfe von StyleVision Power Stylesheets
- Leistungsstarker visueller Function Builder zur Erstellung benutzerdefinierter Funktionen
- Aufrufen der MapForce Benutzeroberfläche und -funktionen über die MapForce API

(ActiveX Control)

- Definition von benutzerdefinierten XSLT 1.0- und 2.0-Bibliotheken
- Unterstützung für XPath 2.0-Funktionen in XSLT 2.0 und XQuery
- Definition von benutzerdefinierten Komponenten/Funktionen mit komplexen Inputs/Outputs
- Unterstützung für quellorientiertes Mapping / Mapping von Mixed Content und Alles-kopieren-Verbindungen
- Automatische Beibehaltung von Mapping-Konnektoren fehlender Nodes/Datenelemente
- Unterstützung für HL7 Version 3.x, da diese XML-Schema-basiert ist

Die **Professional** Edition unterstützt zusätzlich:

- XML-Datenmapping von/auf Datenbanken - IBM DB2 und andere
- Direkte Abfrage von Datenbanken
- SQL-WHERE-Filter und SQL-Anweisungsassistent
- SQL SELECT-Anweisungen als Datenquellen für das Mapping
- Integration benutzerdefinierter C++, Java- und C#-Funktionen
- Projektverwaltungsfunktionen zum Gruppieren von Mappings
- MapForce Plug-in für Eclipse 3.4 / 3.5 / 3.6
- MapForce für Microsoft Visual Studio
- Dokumentation des Mapping-Designs

Die **Enterprise** Edition unterstützt zusätzlich:die Anzeige

- Erstellung von SOAP 1.1 und SOAP 1.2 Webserviceprojekten und Mapping von Webservice-Operationen von WSDL 1.1 und WSDL 2.0-Dateien
- Direkter Aufruf von Webservice-Funktionen
- FlexText™: verbesserte Verarbeitung von Altdatenbeständen (Legacy-Dateien)

Alle Transformationen lassen sich in ein und derselben Arbeitsumgebung durchführen, in der mehrere Quellen und Ziele gemischt werden können und umfassende und erweiterbare Funktionsbibliotheken für alle Arten von Datenmanipulationen zur Verfügung stehen.

3.1 Terminologie

Im Folgenden sind die in dieser Dokumentation verwendeten Begriffe definiert.

Bibliothek

Eine Bibliothek ist eine Sammlung von Funktionen, die im Bibliotheksfenster zu sehen sind. Es gibt verschiedene Funktionstypen, Basisfunktionen (Core) und computersprachspezifische sowie benutzerdefinierte Funktionen. Nähere Informationen dazu finden Sie im Abschnitt "[Funktionen](#)".

Komponente

In MapForce werden viele der grafischen Elemente, die Sie auf der Mapping-Registerkarte einfügen/importieren oder platzieren können, zu Komponenten. Komponenten haben kleine **Dreiecke**, über die Sie Daten zwischen Quell- und Zielkomponenten mappen können, indem Sie Verbindungen zwischen diesen ziehen.

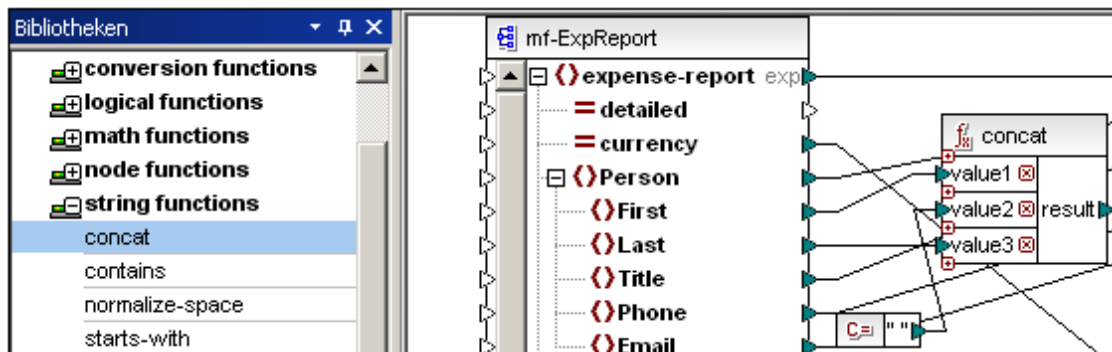
Komponenten erkennt man an den kleinen **Dreiecken**, die sie besitzen. Mit Hilfe dieser Dreiecke (**Input-** und **Output-Symbole**) können Sie Daten aufeinander mappen, indem Sie eine Verbindung zwischen ihnen erstellen.

Die folgenden Dateien werden zu Komponenten, wenn Sie in den Mapping-Bereich platziert werden:

- Schemas und DTDs: Quell- und Zielschemas
- Datenbanken: Quell- und Zieldatenbanken
- Flat Files: CSV- und andere Textdateien
- EDI-Dokumente (UN/EDIFACT und ANSI X12 und HL7): Quell- und Zieldokumente
- Excel 2007 (und höher)-Quell- und Zieldateien
- XBRL-Dokumente, Quelle und Ziele
- Funktionstypen: XSLT/XSLT2, XQuery, Java-, C#-, and C++-Funktionen sowie Konstanten, Filter und Bedingungen

Funktion

Bei einer Funktion handelt es sich um eine vordefinierte Komponente, die Datenoperationen durchführt. Datenoperationen wie z.B. **Concat**. Funktionen haben **Eingabe-** und/oder **Ausgabeparameter**, wobei es für jeden Parameter ein eigenes Input/Output-Symbol gibt. Die Funktionen stehen im Bibliotheksfenster zur Verfügung und sind nach logischen Kriterien gruppiert. Wenn Sie Strg+F drücken, können Sie nach einer Funktion suchen. Wenn Sie eine Funktion in das Mapping-Fenster ziehen, wird eine Funktionskomponente erstellt. Nähere Informationen dazu finden Sie im Abschnitt [Funktionen und Bibliotheken](#).



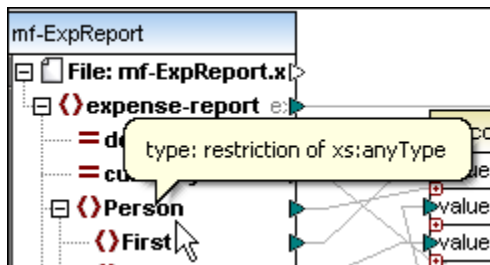
Java ausgewählt

Datenelement

Ein Datenelement ist eine Dateneinheit, die von einer Komponente auf eine andere gemappt

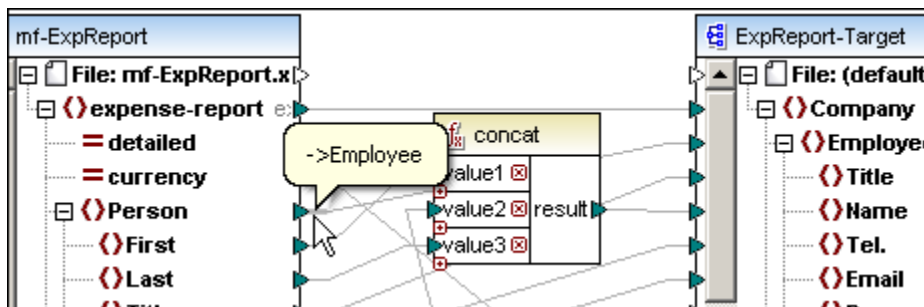
werden kann. Es kann sich dabei um ein **Element**, ein **Attribut**, ein Datenbankfeld, ein EDI-Segment, ein Excel-Feld oder ein XBRL-Element handeln.

Jedes Datenelement verfügt über ein **Input-** und ein **Output-Symbol**. Um ein Mapping zwischen zwei Datenelementen zu erzeugen, muss es sich **nicht** um denselben Typ (Element oder Attribut) handeln.



Eingabe-, Ausgabesymbol

Die kleinen Dreiecke neben den Komponenten sind Input- und Output-Symbole. Durch Anklicken und Ziehen eines Symbols erstellen Sie einen **Konnektor**, der eine Verbindung zu einem anderen Symbol herstellt, sobald Sie die Maustaste loslassen. Der Konnektor **kennzeichnet ein Mapping** zwischen den zwei durch die Symbole markierten Daten-Sets. Nähere Informationen dazu finden Sie im Abschnitt "[Mapping zwischen Komponenten](#)".



Konnektor

Der Konnektor ist die Verbindungslinie zwischen zwei Symbolen. Er kennzeichnet das Mapping zwischen den beiden durch die Symbole markierten Datensätzen. Nähere Informationen dazu finden Sie im Abschnitt "[Mapping zwischen Komponenten](#)".

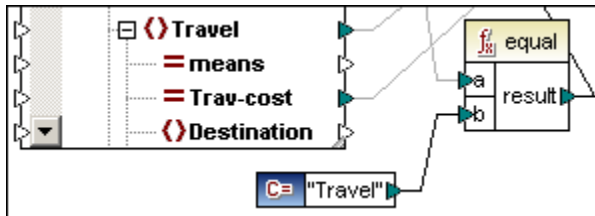
Es können mehrere Arten von Konnektoren definiert werden:

- Zielorientierte (Standard) Konnektoren, siehe: [Quellorientiertes/Mixed Content und Standard-Mapping](#)
- "Alles kopieren" Konnektoren, siehe: ["Alles kopieren" Konnektoren](#)
- Quellorientierte (Mixed Content) Konnektoren, siehe [Quellorientiertes und Mixed Content Mapping](#)



Konstante

Eine Konstante ist eine Komponente, die für ein Input-Symbol einer Funktion oder Komponente konstante Daten zur Verfügung stellt. So ist z.B. der String "Travel" mit dem Parameter "b" der equal-Funktion verbunden. Beim Erstellen der Komponente oder wenn Sie darauf doppelklicken werden diese Daten in ein Dialogfeld eingegeben. Eine Konstantenfunktion hat nur ein Output-Symbol. Sie können zwischen den folgenden Datentypen wählen: String, Zahl und Alle anderen (String).



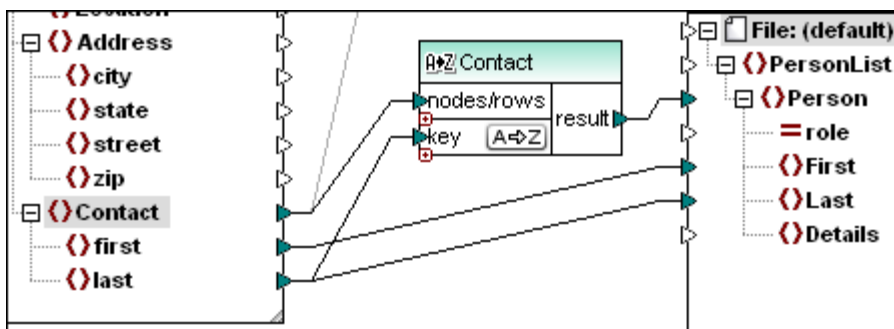
Variable

Fügt eine Zwischenvariable ein, die einer regulären (nicht-inline gesetzten) benutzerdefinierten Funktion entspricht. Variablen sind Strukturkomponenten ohne Instanzdateien und dienen zum Vereinfachen eines Mappings.



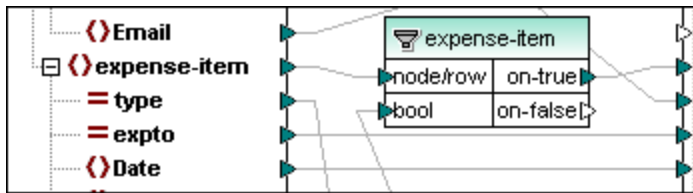
Sortierkomponente

Eine Sortierkomponente sortiert die Input-Daten nach einem bestimmten von Ihnen definierten/gemappten Schlüssel. Die Sortierreihenfolge kann durch Klicken auf die Schaltfläche A=>Z im Komponentenparameterfeld "key" geändert werden.



Filter: Node/Zeile

Ein Filter ist eine Komponente, die Daten über zwei Eingabe- und Ausgabeparameter filtert: **node/row** und **bool** und **on-true**, **on-false**. Ist der Boolesche Wert "true", wird der Wert/Inhalt des node/row-Parameters in den **on-true**-Parameter übernommen.



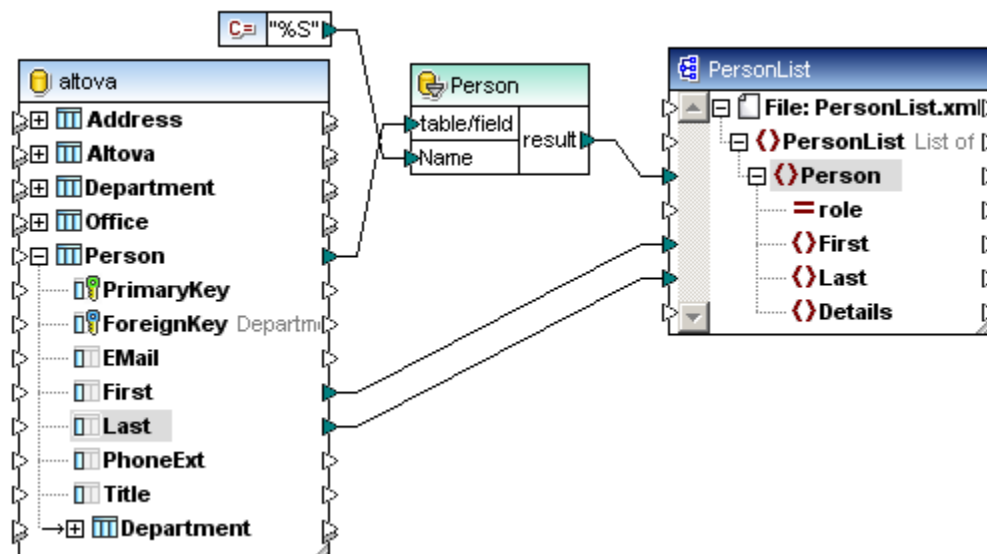
Der **on-false**-Ausgabeparameter gibt das vom Mapping definierte Komplement-Node-Set aus. Nähere Informationen dazu finden Sie unter [Mehrere Zielschemas / -dokumente](#).



SQL-WHERE/ORDER-Bedingung

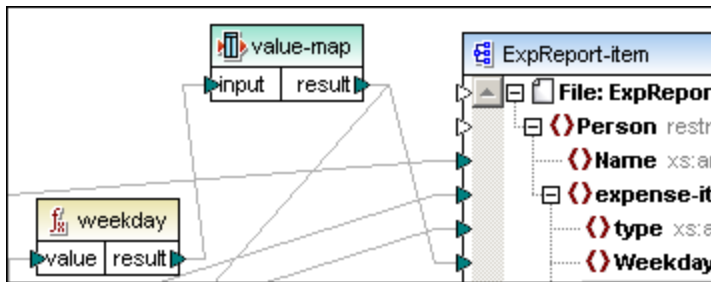
Mit Hilfe der SQL-WHERE/ORDER-Komponente können Sie Datenbankdaten mit Bedingungen filtern. Wenn Sie auf die Komponente doppelklicken, können Sie die SQL WHERE/ORDER-Anweisung eingeben. Die SQL WHERE/ORDER-Komponente besteht aus mehreren Teilen:

- der **Select**-Anweisung, die beim Herstellen der Verbindung zu einer Datenbanktabelle automatisch generiert wird
- der **WHERE**-Klausel, die Sie manuell in das SQL WHERE-Auswahlfeld eingeben. Beachten Sie, dass die Sekundärschlüssel automatisch in die Select-Anweisung inkludiert werden.
- der **ORDER BY**-Klausel



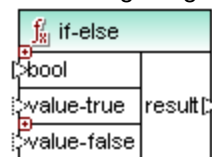
Wertezuordnung

Mit Hilfe der Wertezuordnungskomponente können Sie eine Gruppe von Eingabedaten anhand einer Lookup-Tabelle in eine andere Gruppe von Ausgabedaten transformieren. Wenn Sie auf die Komponente doppelklicken, wird die Wertezuordnungstabelle geöffnet. In der linken Tabellenspalte wird der Input definiert, in der rechten Spalte sind die transformierten Daten, die ausgegeben werden sollen, definiert.



IF-Else-Bedingung

Eine Bedingung ist eine Komponente, die gestattet, je nach Ergebnis einer vordefinierten Situation unterschiedliche Daten-Sets weiterzugeben. Im Titel der Komponente wird der Text **if-else** angezeigt. Ein Beispiel finden Sie im Referenzabschnitt unter "[Bedingung](#)".



- Der erste Eingabewert ist ein **Boolscher** Wert, der die Daten enthält, auf die hin der Datensatz überprüft wird.
- Der Eingabeparameter **value-true** enthält die Daten, die übernommen werden sollen, wenn die Bedingung zutrifft, also "true" ist.
- Der Parameter **value-false** enthält die Daten, die übernommen werden sollen, wenn die Bedingung nicht zutrifft, also "false" ist.
- Der Parameter **result** gibt die Daten der Eingabeparameter value-true/false aus.

Die IF-Else-Funktion ist nun **erweiterbar**. Das bedeutet, dass überprüft werden kann, ob mehrere IF-Werte vorhanden sind. Die Else-Bedingung/der Else-Wert kann über den Parameter **otherwise** ausgegeben werden. Nähere Informationen dazu finden Sie unter [Einfügen | If-Else](#).

Kapitel 4

MapForce Tutorial

4 MapForce Tutorial

In diesem Tutorial erhalten Sie anhand einer Reihe von Aufgaben einen Überblick darüber, wie Sie die Funktionalitäten von MapForce 2012 am besten einsetzen können.

Ziel dieses Tutorials ist eine einfache Reisekostenabrechnung eines Mitarbeiters auf eine komplexere Firmenspesenabrechnung zu mappen. In unserem Tutorial-Beispiel füllt jeder Mitarbeiter die Felder seiner persönlichen Reisekostenabrechnung aus. Diese Abrechnung wird auf die Firmenspesenabrechnung gemappt und in die Verwaltungsabteilung geroutet. Nun müssen zusätzliche Daten im Zusammenhang mit dem Mitarbeiter eingegeben werden. Das Ergebnis ist eine standardisierte Firmenspesenabrechnung.

In diesem Tutorial lernen Sie Folgendes:

- Einrichten der Mapping-Umgebung
- Mappen der XML-Quelldatei (die persönliche Reisekostenabrechnung) auf die Zielkomponente (die Firmen-Spesenabrechnung)
- Anwendung von Filtern auf die Quelldaten
- Generierung einer XSLT-Transformationsdatei
- Transformation der Quelldaten in die Zieldaten mit Hilfe der generierten XSLT-Datei

Installation und Konfiguration

Es wird in diesem Tutorial davon ausgegangen, dass Sie MapForce erfolgreich auf Ihrem Rechner installiert haben und einen kostenlosen Evaluierungs-Keycode erhalten haben bzw. ein registrierter Benutzer des Produkts sind. Die Evaluierungsversion von MapForce hat den vollen Funktionsumfang, ist jedoch auf einen 30-Tage-Zeitraum befristet. Sie können eine reguläre Lizenz über unseren sicheren Webserver oder einen unserer Vertriebspartner erwerben.

Tutorial-Beispieldateien

In diesem Tutorial wird mit den folgenden Komponenten gearbeitet:

- Quell- und (mehrere) Zielschemas
- einer MS Access-Datenbank als Datenquelle
- verschiedenen Funktionen u.a.: concat, filter, equal und Konstanten

Sie finden alle in diesem Tutorial verwendeten Dateien im Ordner **C:\Documents and Settings\All Users\Application Data\Altova**. Wenn die Applikation von einem Benutzer zum ersten Mal gestartet wird, werden die Beispieldateien für diesen Benutzer in den Ordner [...\MapForceExamples\Tutorial\](#) kopiert. Daher sollten die Beispieldateien im ursprünglichen Verzeichnis [...\All Users\...](#) nicht verschoben, bearbeitet oder gelöscht werden.

Mitinstalliert werden auch die XSLT-Dateien und die transformierten XML-Dateien. Im Tutorial werden die folgenden Dateien verwendet:

Persönliche Reisekostenabrechnung:

- Tut-ExpRep Das Mapping für die Reisekostenabrechnung (ein einziges Ziel)
ort.mfd
- Tut-ExpRep Die Ziel-Spesenabrechnung, basierend auf mehreren Schemas
ort-multi.mfd
- PersonDB.mDas Mitarbeiter-Mapping unter Verwendung einer MS Access-Datenbank
fd as Datenquelle
- mf-ExpRepo Persönliche Reisekostenabrechnung als XML-Instanzdokument
rt.xml

- mf-ExpRepo Verknüpfte Schemadatei
rt.xsd

Firmen-Spesenabrechnung:

- ExpReport-Target.x ml XML-Instanzdokument für die Firmen-Spesenabrechnung
Verknüpfte Schemadatei
- ExpReport-Target.xsd

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-Version auf 64-Bit OS	C:\Programme (x86)\Altova

4.1 Einrichten der Mapping-Umgebung

In diesem Abschnitt wird erklärt, wie man das Quell- und Zielschema für das Mapping definiert.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man die Mapping-Umgebung in MapForce einrichtet. Insbesondere werden folgende Schritte beschrieben:

- Erstellen der Quell- und Zielschemakomponente
- Definieren der XML-Quelldatei
- Auswählen des Root-Elements des Zielschemas

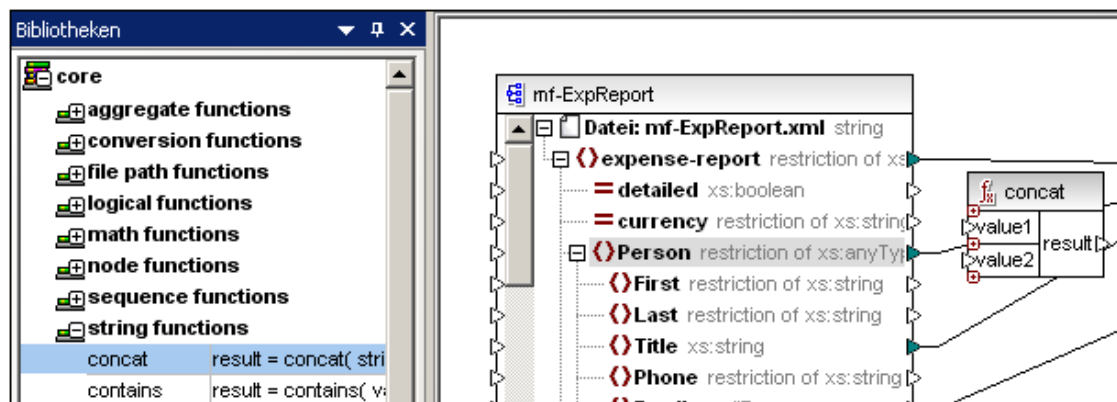
In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.

Bitte beachten Sie:

Die Bildunterschriften unterhalb der Abbildungen zu den einzelnen Bibliotheksfenstern (hier z.B. **XSLT ausgewählt**) geben an, welche Ziel-/Ausgabesprache gerade verwendet wird, wenn Sie zur Vorschau auf das Mapping auf das Register "Ausgabe" klicken. Die ausgewählte Sprache bestimmt auch, welche Funktionen im Bibliotheksfenster angezeigt werden.

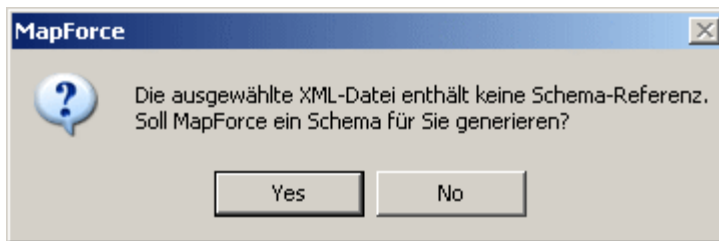


XSLT ausgewählt

4.1.1 Hinzufügen von Komponenten zum Mapping-Fenster

Nachdem Sie MapForce gestartet haben, müssen Sie die Quell- und Zieldateien in einem Mapping-Fenster hinzufügen; Sie können die Dateien auch hinzufügen, indem Sie sie mit der Maus aus dem Windows Explorer in das Mapping-Fenster ziehen.

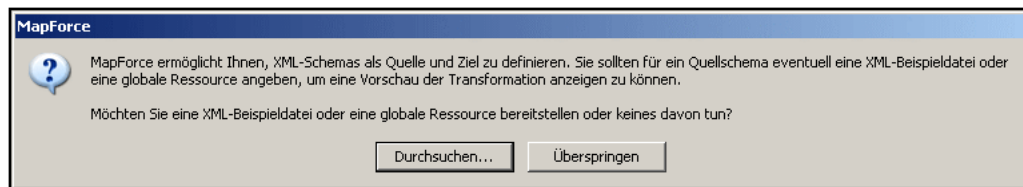
MapForce kann anhand einer bestehenden XML-Datei automatisch ein XML-Schema generieren, wenn kein XML-Schema vorhanden ist. Beim Einfügen einer XML-Datei über den Menübefehl "XML-Schema einfügen" aus dem Menü "Datei" erscheint automatisch ein Dialogfeld, in dem Sie gefragt werden, ob ein XML-Schema erstellt werden soll, wenn keine dazugehörige XML-Schema-Datei gefunden wird.



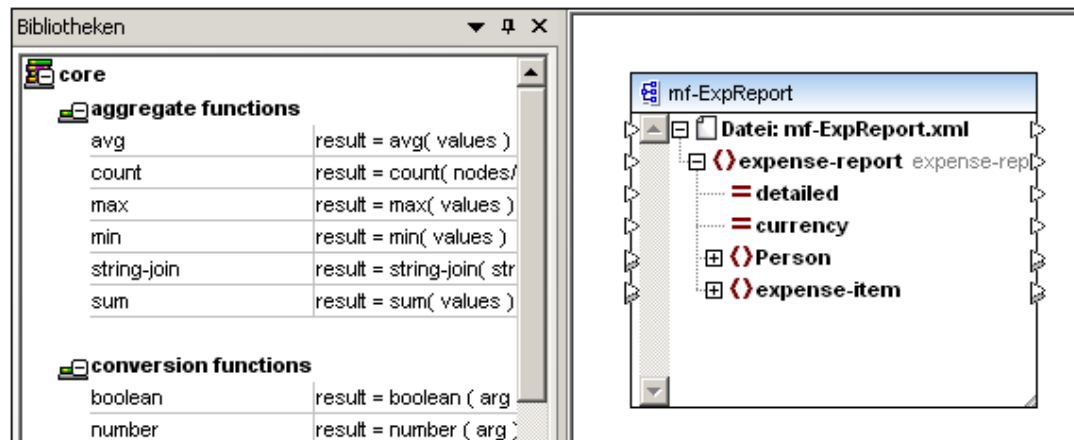
Bei Generierung eines Schemas anhand einer XML-Datei müssen die Datentypen für Elemente/Attribute anhand des XML-Instanzdokuments abgeleitet werden und entsprechen eventuell nicht genau Ihren Erwartungen. Überprüfen Sie bitte, ob das generierte Schema tatsächlich die Instanzdaten genau darstellt.

Erzeugen der Quellschema-Komponente:

1. Klicken Sie auf die Schaltfläche **XML-Schema/Datei einfügen**  oder wählen Sie den Menübefehl **Einfügen | XML-Schema/Datei...**
2. Wählen Sie im Dialogfeld "Öffnen" unter ...\\MapForce2012\\MapForceExamples im Unterordner "Tutorial" die Datei mf-ExpReport.xsd. Sie werden nun aufgefordert, eine XML-Beispieldatei anzugeben, aus der die Daten für den Vorschaubereich verwendet werden sollen.

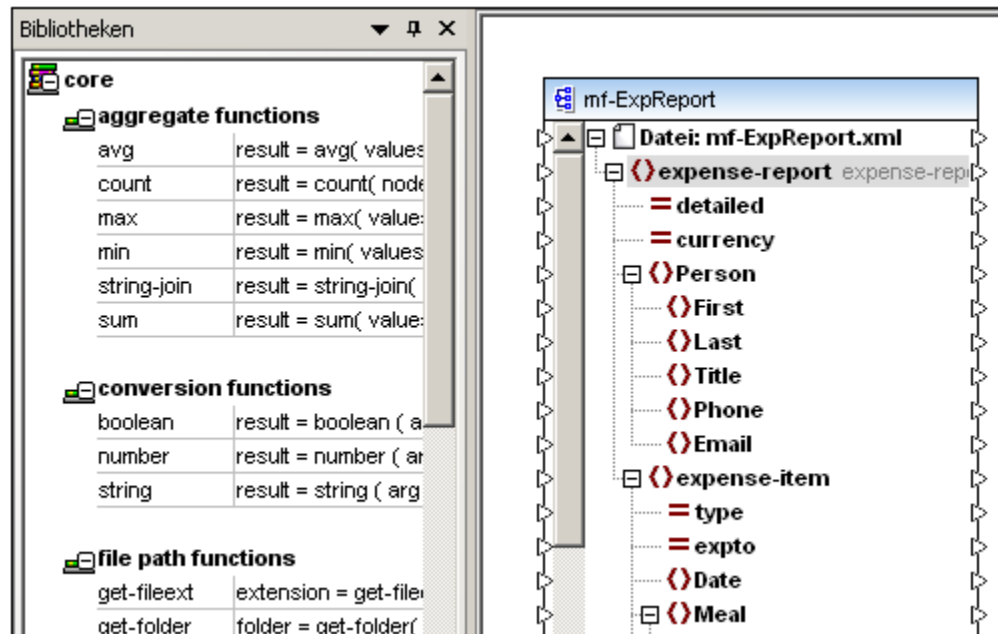


3. Klicken Sie auf die Schaltfläche **Durchsuchen...** und wählen Sie die Datei **mf-ExpReport.xml**. Die Quellschemakomponente wird nun im Mapping-Fenster angezeigt.



XSLT ausgewählt

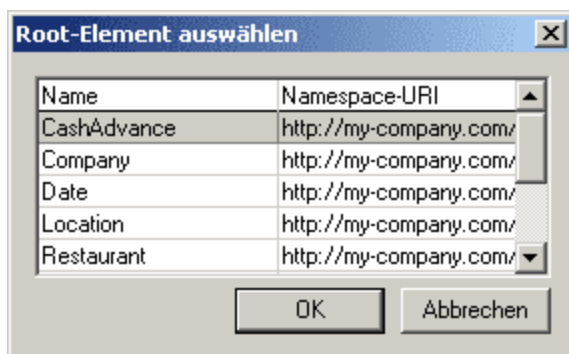
4. Klicken Sie auf den Eintrag **expense-report** der Komponente und drücken Sie auf dem numerischen Tastenblock die *-Taste, um alle Datenelemente anzuzeigen.
5. Klicken Sie auf das Symbol zum Anpassen der Größe  in der rechten unteren Ecke der Komponente und ziehen Sie es, um die Fenstergröße anzupassen.
Anmerkung: Wenn Sie auf das Symbol doppelklicken, erhält die Komponente die "**optimale Größe**", so dass alle Elemente zu sehen sind.



XSLT ausgewählt

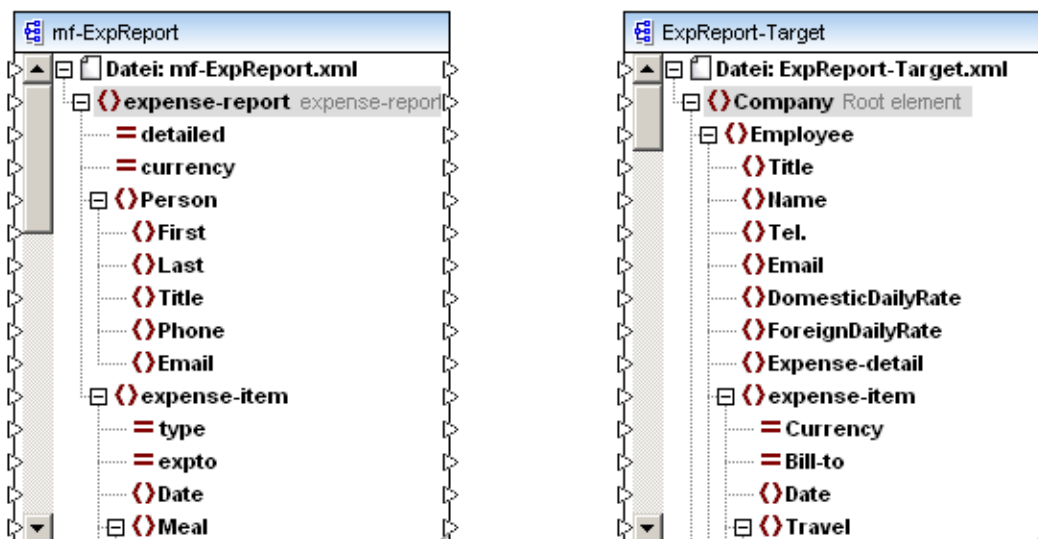
Erzeugen der Zielschema-Komponente:

1. Klicken Sie auf das Symbol **XML-Schema/Datei einfügen** oder wählen Sie den Menübefehl **Einfügen | XML-Schema/Datei...**
2. Wählen Sie im Dialogfeld "Öffnen" die Datei **ExpReport-Target.xsd**.
Sie werden nun aufgefordert, eine XML-Beispieldatei für das Schema anzugeben.
3. Klicken Sie auf die Schaltfläche **Überspringen** und wählen Sie **Company** als Root-Element des Zieldokuments aus.



Die Zielschemakomponente wird nun auf dem Mapping-Register angezeigt.

4. Klicken Sie auf den Eintrag **Company** und drücken Sie die *-Taste auf dem numerischen Tastenblock, um alle Datenelemente anzuzeigen.
5. Doppelklicken Sie auf das Kicken Sie auf das Symbol zum Anpassen der Größe und passen Sie die Fenstergröße an.



Wir können nun mit dem Mappen der Schema-Elemente vom Quell- auf das Zielschema beginnen.

Anmerkung: Wenn Sie eine Komponente mit der Maus ziehen, werden Hilfslinien zur [automatischen Ausrichtung](#) der Komponente angezeigt, um die Komponente besser positionieren zu können.

4.2 Erstellen eines Mappings

Im vorhergehenden Abschnitt haben Sie die [Quell- und Zielschemakomponente Ihres Mappings definiert](#). Wir beginnen nun mit dem Mappen der eigentlichen Daten.

Ziel

Sie lernen, wie man die Quell- und Zielkomponente aufeinander mappt und wie man das Mapping-Ergebnis mit Hilfe von Funktionen und Filtern modifizieren kann.

- [Mappen von Schema-Datenelementen](#) über Konnektoren
- Kombinieren von Elementen der Quelldaten [mit Hilfe einer concat-Funktion](#)
- [Filtern von Quelldaten](#), um nur bestimmte Spesen in den Zielbericht zu übernehmen

In diesem Abschnitt verwendete Befehle



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identischer Sub-Einträge ein- bzw. auszuschalten.



Konstante einfügen: Klicken Sie auf diese Schaltfläche, um eine Konstantenkomponente in das gerade aktive Mapping-Fenster einzufügen.



Filter: Nodes/Zeilen: Klicken Sie auf diese Schaltfläche, um eine Filterkomponente in das gerade aktive Mapping-Fenster einzufügen.

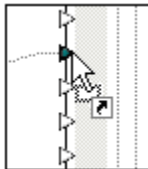
Altova Website:  [Mappen von Daten - Datenintegration](#) und [XML mapping](#)

4.2.1 Mappen von Schema-Elementen

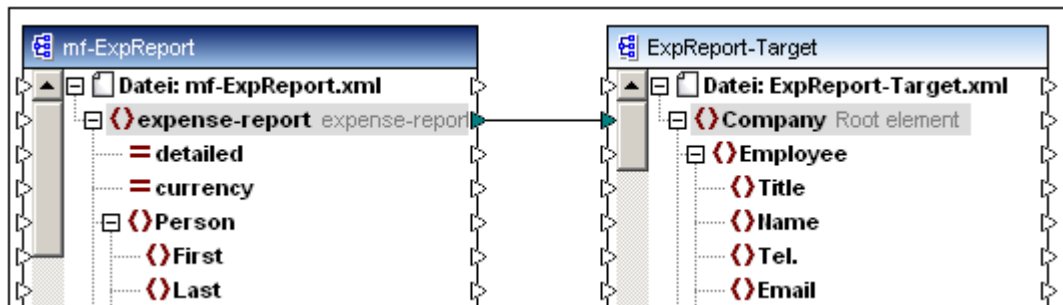
In diesem Abschnitt wird erklärt, wie das Mapping zwischen den Elementen des Quell- und des Zielschemas definiert wird.

So mappen Sie die Schemas mf-ExpReport und ExpReport-Target:

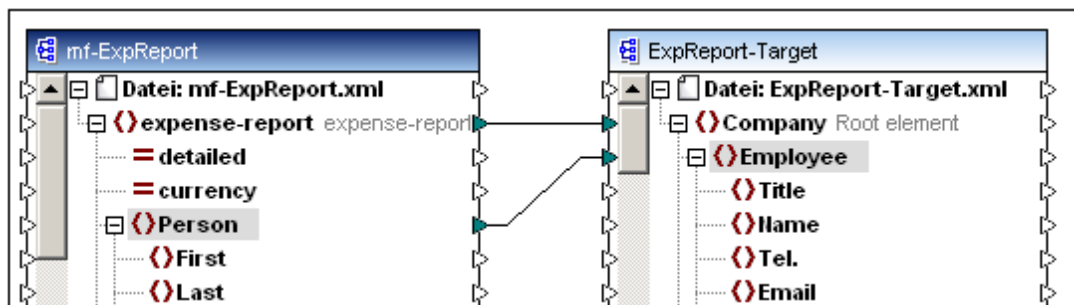
1. Klicken Sie im Quellschema mf-ExpReport auf das "Element" **expense-report** und ziehen Sie es mit der Maus. Vom Output-Symbol wird automatisch eine Verbindungslinie zum Mauszeiger, dessen Symbol sich nun geändert hat, erstellt.
2. Ziehen Sie den Mauszeiger in die Nähe des "Elements" **Company** im Schema ExpReport-Target und lassen Sie die Maustaste los, sobald als Mauszeiger wieder ein Pfeilsymbol angezeigt wird. Unterhalb des Mauszeigers wird ein kleines Verbindungssymbol angezeigt. Wenn diese Aktion zulässig ist, erscheint das Input-Symbol und der Name des Datenelements in der Komponente markiert.



Zwischen Quell- und Zielschema ist nun ein Konnektor gesetzt und ein Mapping erzeugt worden.

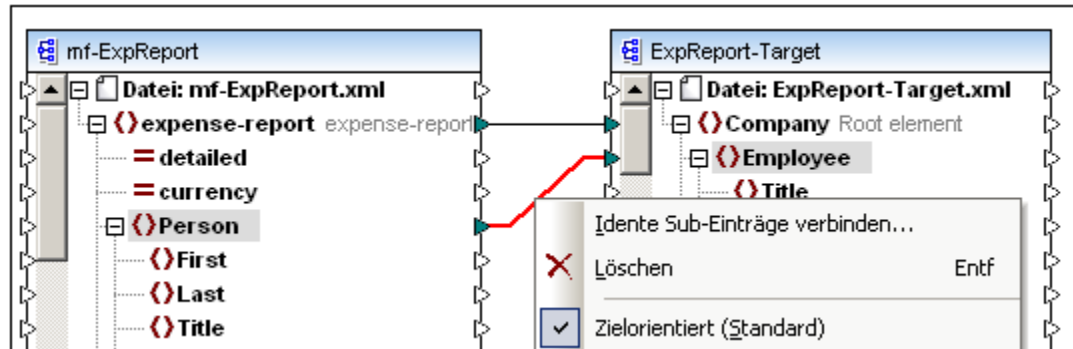


3. Mappen Sie nun die Elemente Person und Employee auf die gleiche Art und Weise.

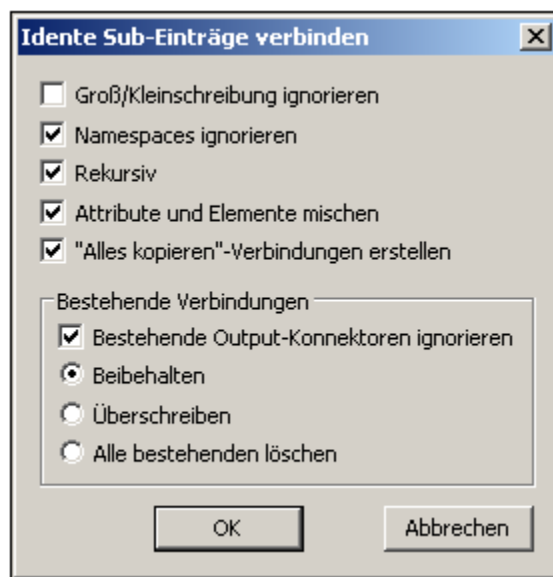


Wenn das Symbol "Sub-Einträge automatisch verbinden" aktiv ist, werden die Datenelemente "Title" und "Email" ebenfalls automatisch verbunden. Falls dies nicht der Fall ist:

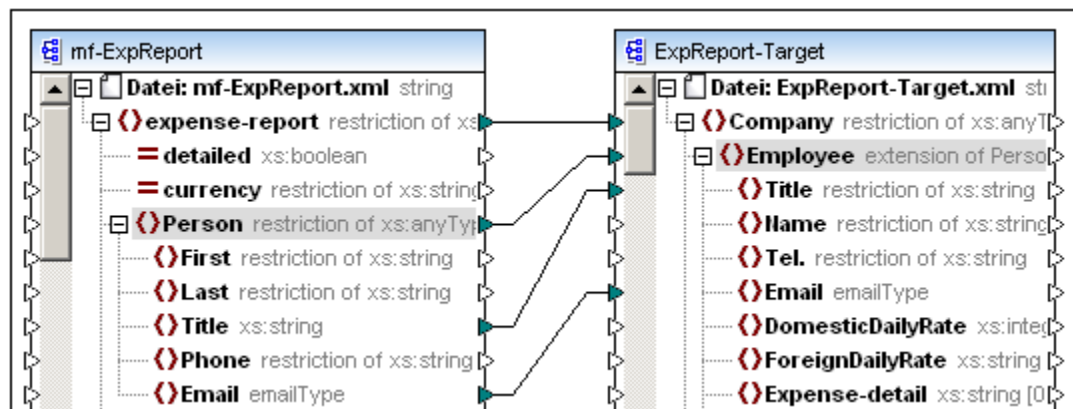
4. Rechtsklicken Sie auf den "Person"-Konnektor und wählen Sie im Kontextmenü die Option "Identische Sub-Einträge verbinden".



Daraufhin wird das Dialogfeld **Identische Sub-Einträge verbinden** geöffnet.

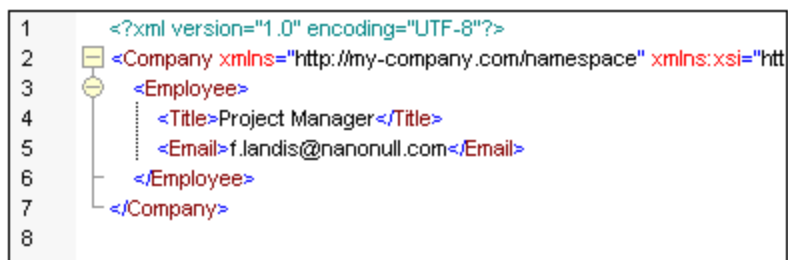


5. Aktivieren Sie die Kontrollkästchen wie in der Abbildung oben und klicken Sie auf OK. Nähere Informationen finden Sie im Abschnitt zu [Konnektoreigenschaften](#).



Für die "Elemente" **Title** und **Email** in den beiden Schemas wurden automatisch Mappings erzeugt.

6. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis im Ausgabefenster zu sehen.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Company xmlns="http://my-company.com/namespace" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://my-company.com/namespace my-company.xsd">
3   <Employee>
4     <Title>Project Manager</Title>
5     <Email>f.landis@nanonull.com</Email>
6   </Employee>
7 </Company>
8
```

Sie werden feststellen, dass die Felder Title und Email Daten enthalten, die aus dem XML-Instanzdokument stammen.

7. Klicken Sie auf die Schaltfläche "Mapping", um wieder in das Mapping-Fenster zurückzuwechseln und mit dem Mapping fortzufahren.

Bitte beachten Sie: Die Einstellungen im Dialogfeld "Identische Sub-Einträge verbinden" werden so lange beibehalten, bis Sie sie ändern. Sie können diese Einstellungen auf eine Verbindung anwenden, indem Sie entweder das Kontextmenü verwenden, oder auf das Symbol "[Identische Sub-Einträge automatisch verbinden](#)" klicken, um diese Option zu aktivieren bzw. zu deaktivieren.

4.2.2 Mappen von Daten mit Hilfe von Funktionen

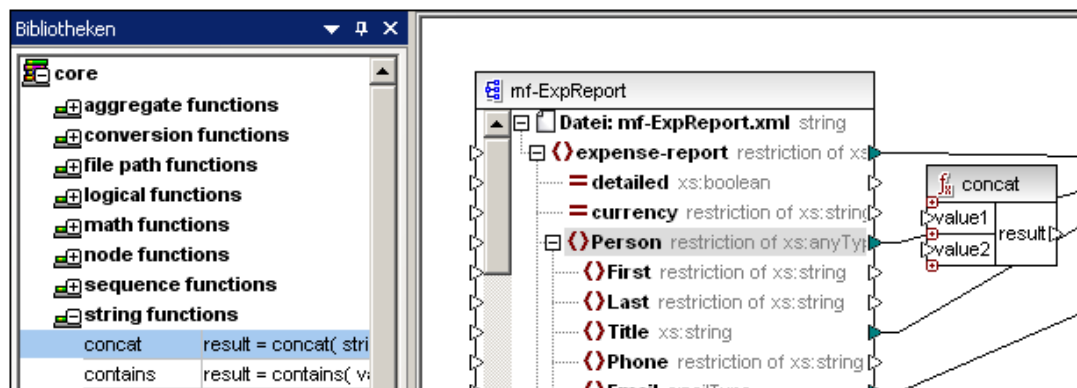
Ziel dieses Abschnitts ist es, zwei Daten-Sets aus dem Quellschema zu verbinden und das Ergebnis im Zieldokument in einem einzigen Datenelement zusammenzufassen. Dies wird in den folgenden Schritten durchgeführt:

- Einsatz der String-Funktion "**concat**", um die Quellschema-Elemente **First** und **Last** zu kombinieren
- Verwendung einer **Konstantenfunktion**, um zur Trennung der beiden Elemente ein Leerzeichen einzufügen
- Setzen des Ergebnisses dieses Vorgangs in das Zielschema-Element **Name**.

Beachten Sie bitte, dass einige der zuvor definierten Mappings aus Gründen der Übersichtlichkeit in den folgenden Abbildungen nicht zu sehen sind.

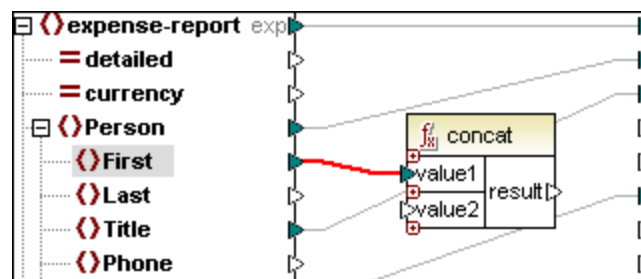
Kombinieren von Datenelementen mit Hilfe von Funktionen:

1. Erweitern Sie auf dem Register "Bibliotheken" unter **core** die Funktionsgruppe **string functions**, klicken Sie auf den Eintrag **concat** und ziehen Sie ihn in das Mapping-Fenster.

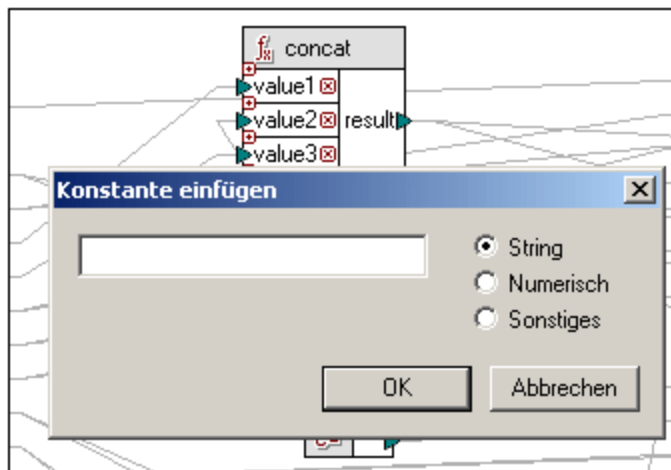


XSLT ausgewählt

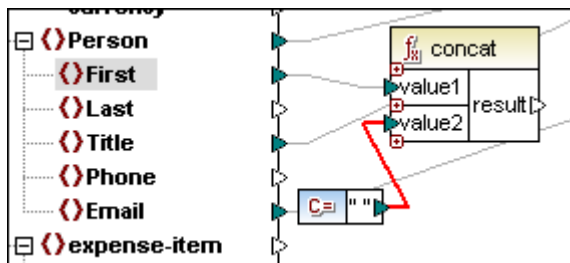
2. Wählen Sie in der Komponente "mf-ExpReport" das Datenelement **First** aus, halten Sie die Maustaste gedrückt und ziehen Sie den Cursor auf den Input **value1** der concat-Komponente, um eine Verbindung herzustellen.



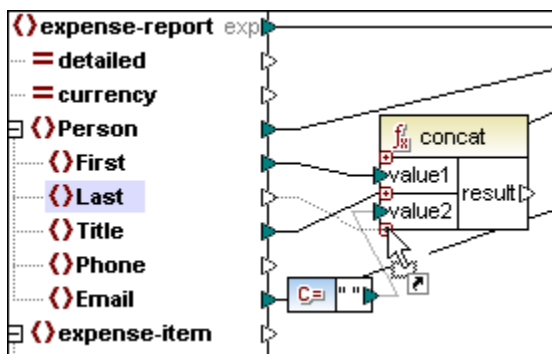
3. Klicken Sie in der Symbolleiste mit der rechten Maustaste auf den Hintergrund in der Nähe von **value2** und wählen Sie im Kontextmenü den Befehl **Konstante einfügen**, um eine Konstantenkomponente einzufügen.



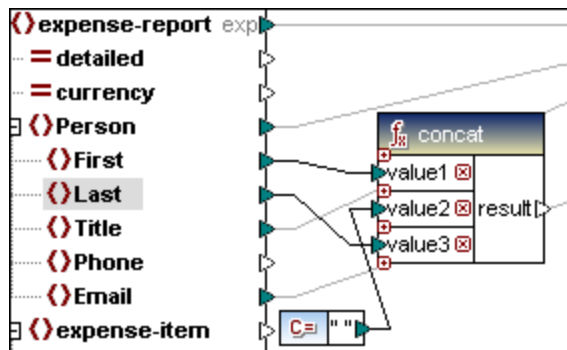
4. Geben Sie im Textfeld ein Leerzeichen ein und klicken Sie auf **OK**. Die Konstantenkomponente befindet sich nun im Arbeitsbereich und ihr Inhalt wird neben dem Output-Symbol angezeigt.
5. Erstellen Sie eine Verbindung zwischen der Komponente **constant** und **value2** der concat-Komponente.



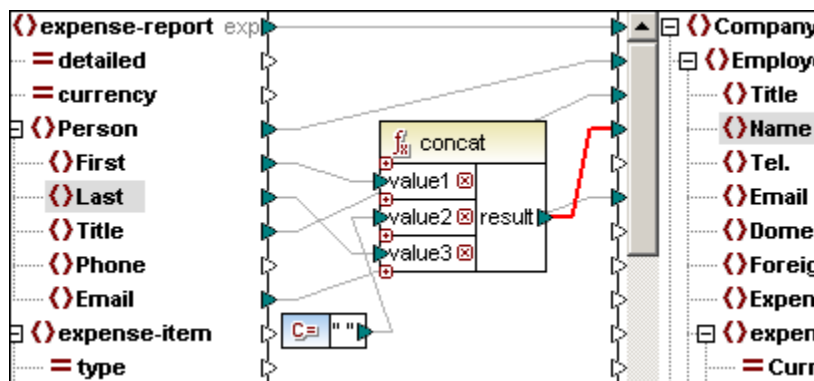
6. Klicken Sie in der Komponente "mf-ExpReport" auf das Datenelement **Last** und ziehen Sie den Konnektor auf das "+"-Symbol der concat-Funktion unterhalb von **value2**. Der Mauszeiger ändert sich und zeigt an, an welcher Stelle Sie den Konnektor platzieren können.



Dadurch wird die concat-Funktion automatisch um ein weiteres Datenelement (Wert) erweitert, mit dem das Datenelement "Last" verbunden wird.



7. Verbinden Sie das **result**-Symbol der concat-Komponente mit dem Datenelement **Name** im Zielschema.



8. Klicken Sie auf die Schaltfläche **Ausgabe**, um das Ergebnis des aktuellen Mappings im Ausgabefenster zu sehen.

```

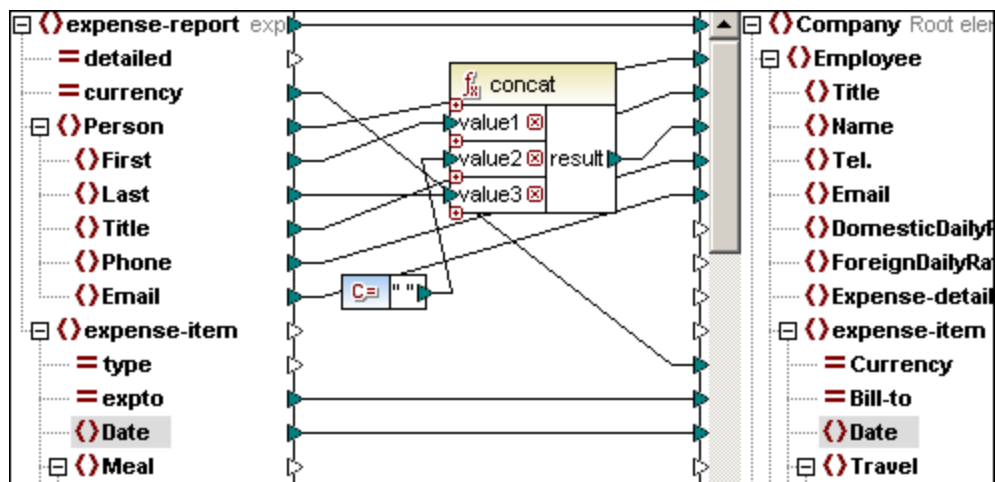
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3  <Employee>
4  <Title>Project Manager</Title>
5  <Name>Fred Landis</Name>
6  <Email>f.landis@nanonull.com</Email>
7  </Employee>
8  </Company>
9

```

Sie werden sehen, dass sich zwischen den **Name**-Tags nun der Personennamen "Fred Landis" befindet. Vor- und Nachname sind durch ein Leerzeichen getrennt.

Mappen der übrigen Personendaten:

- Erzeugen Sie zwischen den folgenden Datenelementen Mappings:
 - currency auf currency
 - Phone auf Tel.
 - expto auf Bill-to
 - Date auf Date



2. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis zu sehen.

```
<?xml version="1.0" encoding="UTF-8"?>
<Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schema
  <Employee>
    <Title>Project Manager</Title>
    <Name>Fred Landis</Name>
    <Tel.>123-456-78</Tel.>
    <Email>f.landis@nanonull.com</Email>
    <expense-item Currency="USD" Bill-to="Sales">
      <Date>2003-01-02</Date>
      <Date>2003-01-01</Date>
      <Date>2003-07-07</Date>
      <Date>2003-02-02</Date>
      <Date>2003-03-03</Date>
    </expense-item>
  </Employee>
</Company>
```

Im Moment stammen fünf Datenelemente aus der zugewiesenen XML-Instanzdatei.

Bitte beachten Sie:

Um Platz auf dem Bildschirm zu sparen können Funktionen in benutzerdefinierten Funktionen/Komponenten gruppiert werden. Ein Beispiel dafür, wie man die concat- und die constant-Funktion zu einer einzigen benutzerdefinierten Funktion/Komponente kombiniert finden Sie unter "[Benutzerdefinierte Funktionen/Komponenten](#)".

4.2.3 Filtern von Daten

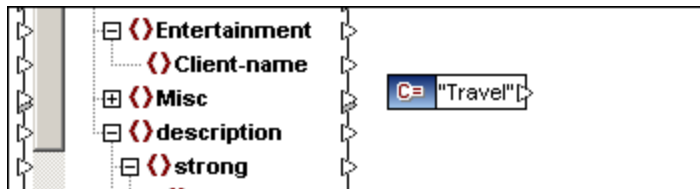
Im diesem Abschnitt sollen die Ausgaben für Unterkunft und Essen herausgefiltert werden, sodass nur die Reisekosten im Zielschema/-dokument übernommen werden.

Dies wird bewerkstelligt durch:

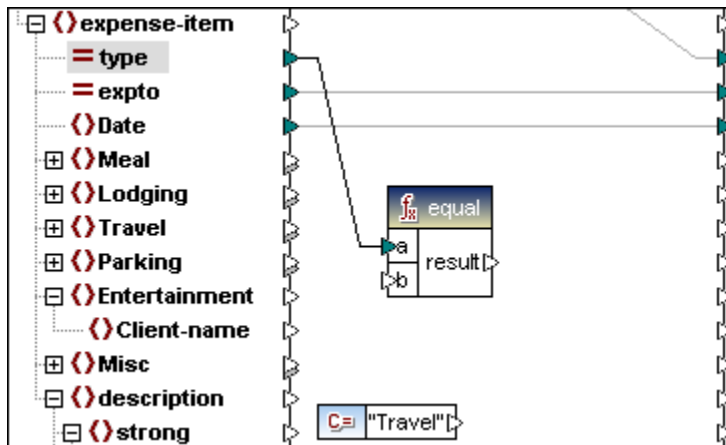
- Verwendung der **equal**-Funktion zum Testen des Werts eines Quelldatenelements
- Verwendung einer **Konstantenfunktion**, um den zu testenden Vergleichsstring anzugeben
- Verwendung der **Filter**-Komponente, die die Reisekostendaten weitergibt, bei denen der Boolesche Eingabewert True ergibt
- Setzen des on-true-Ergebnisses in das Zielschema/-dokument-Element **expense-item**.

Filtern von Daten:

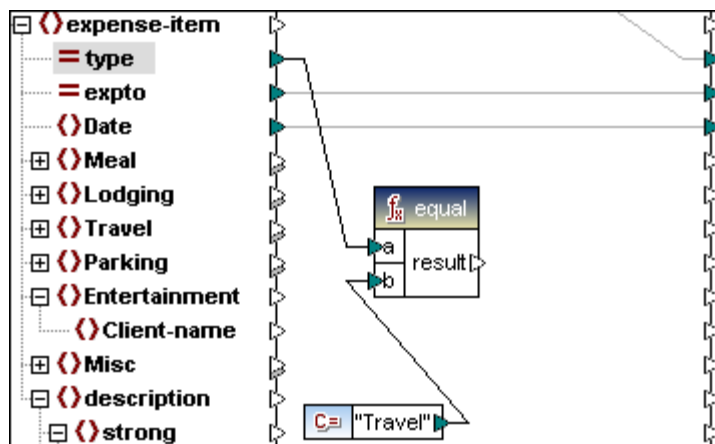
1. Klicken Sie auf die Schaltfläche **Konstante einfügen**  um eine Konstantenkomponente einzufügen und geben Sie in das Eingabefeld den String **Travel** ein.



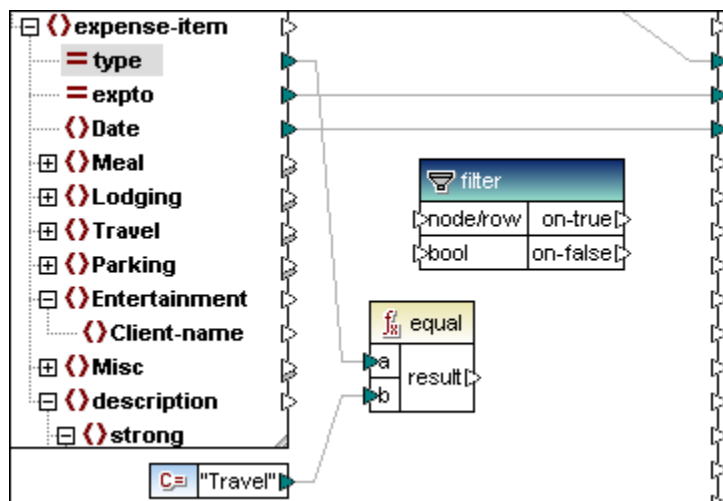
2. Erweitern Sie auf dem Register "Bibliotheken" in der **core**-Bibliothek die Gruppe **logical functions** und ziehen Sie die logische Funktion **equal** in das Mapping Fenster.
3. Verbinden Sie das **expense-item**-Attribut **type** im Quellschema mit dem Parameter **a** der **equal**-Funktion.



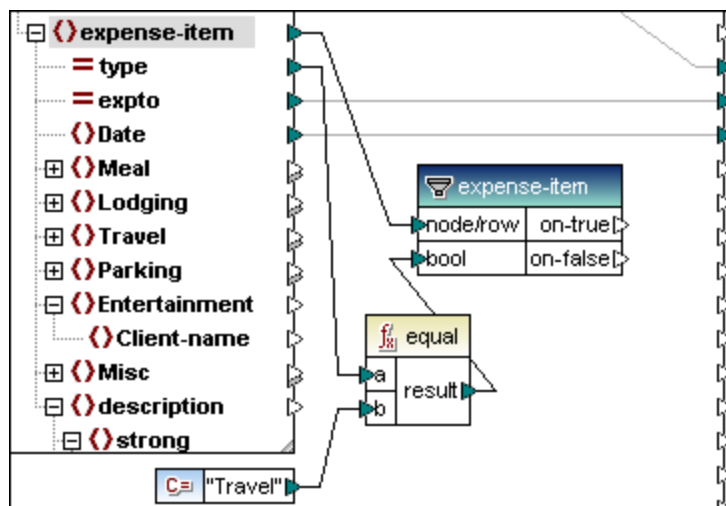
4. Verbinden Sie das **result**-Symbol der Travel-Konstantenkomponente mit dem Parameter **b** der **equal**-Funktion.



5. Wählen Sie die Menüoption **Einfügen | Filter: Nodes/Zeilen**.

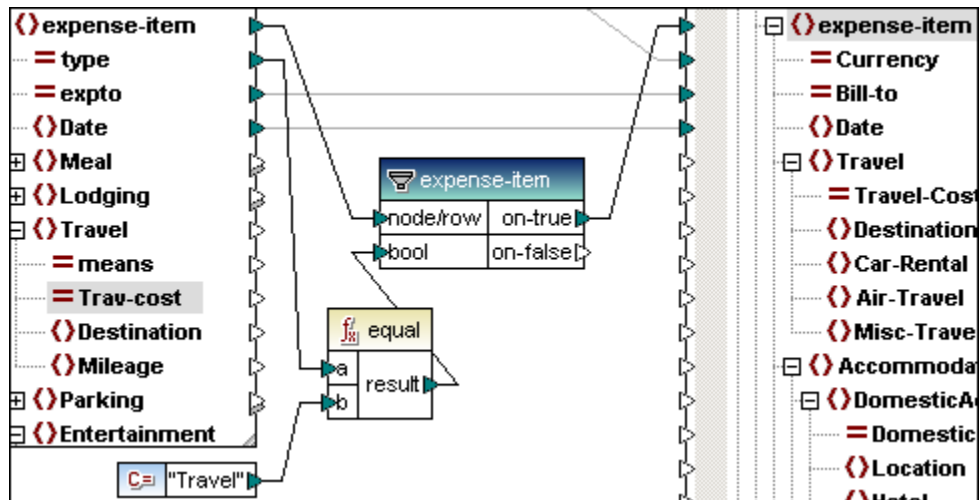


6. Verbinden Sie das **result**-Symbol der **equal**-Komponente mit dem **Bool**-Parameter der **Filter**-Komponente
7. Verbinden Sie das **expense-item**-Symbol des Quellschemas mit dem **node/row**-Parameter der Filterkomponente.

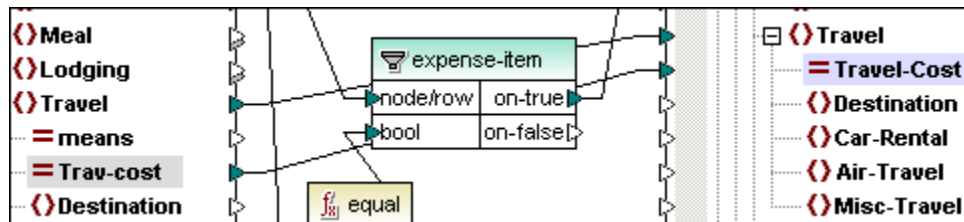


Beachten Sie dass sich der Name der Filterkomponente nun in "expense-item" ändert.

8. Verbinden Sie das **on-true**-Symbol der **Filter**-Komponente mit dem **expense-item**-Element des Zieldokuments.



9. Verbinden Sie das Element **Travel** im Quellschema mit dem Element **Travel** im Zielschema/-dokument.
10. Verbinden Sie das Attribut **Trav-cost** mit dem Attribut **Travel-Cost** im Zielschema/-dokument.



11. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis im Ausgabefenster zu sehen.


```
1      <?xml version="1.0" encoding="UTF-8"?>
2      <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.w3.org/2001/XMLSchema-instance http://www.w3.org/2001/XMLSchema-instance">
3      <Employee>
4          <Title>Project Manager</Title>
5          <Name>Fred Landis</Name>
6          <Tel.>123-456-78</Tel.>
7          <Email>f.landis@nanonull.com</Email>
8          <expense-item Currency="USD" Bill-to="Development">
9              <Date>2003-01-02</Date>
10             <Travel Travel-Cost="337.88"/>
11         </expense-item>
12         <expense-item Currency="USD" Bill-to="Accounting">
13             <Date>2003-07-07</Date>
14             <Travel Travel-Cost="1014.22"/>
15         </expense-item>
16         <expense-item Currency="USD" Bill-to="Marketing">
17             <Date>2003-02-02</Date>
18             <Travel Travel-Cost="2000"/>
19         </expense-item>
20     </Employee>
21 </Company>
22
```

Bitte beachten Sie: Der **on-false**-Parameter der Filterkomponente gibt das **Komplement-Nodeset** aus, das vom on-true-Parameter gemappt wird. In diesem Beispiel wären das alle **nicht-travel**-Reisekostenposten.

Die Anzahl der Expense-item-Elemente hat sich nun auf drei reduziert. Wenn Sie dies mit der Datei **mf-ExpReport.xml** vergleichen, sehen Sie, dass nur die Reisekosten übrig bleiben, die Unterkunfts- und Essensausgaben wurden herausgefiltert.

4.3 Generieren von XSLT 1.0- und 2.0-Code

Nachdem Sie nun das [Mapping erstellt haben](#), können Sie die eigentliche Transformation der Quelldaten durchführen. MapForce kann zwei Versionen von XSLT-Code generieren: XSLT 1.0 und XSLT 2.0.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man eine [Vorschau](#) des XSLT-Codes anzeigt, diesen generiert und speichert und wie man den generierten XSLT-Code ausführt. Insbesondere werden die folgenden Schritte beschrieben:

- Generieren und Speichern von XSLT-Code der gewünschten Art
- Downloaden des Altova XML-Prozessors und Ausführen der Batch-Transformationsdatei

In diesem Abschnitt verwendete Befehle

Datei | Code generieren in: Wählen Sie diese Option, um die Ausgabesprache (d.h. **XSLT 1.0** oder **XSLT 2.0**) auszuwählen. Mit dem Befehl wird das Dialogfeld **Verzeichnis auswählen** geöffnet, wo Sie den Ordner definieren, in dem der generierte XSLT-Code gespeichert werden soll.

So generieren Sie XSLT-Code:

1. Wählen Sie die Menüoption **Datei | Code generieren in | XSLT 1.0 (bzw. XSLT 2.0)**
2. Geben Sie den Ordner an, in dem die generierte XSLT-Datei abgelegt werden soll und bestätigen Sie dies mit OK.
Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
3. Navigieren Sie zum gewünschten Ordner, wo Sie die XSLT-Datei unter dem Namen **MappingMapToExpReport-Target.xslt** finden. (d.h. in der Form: MappingMapTo<TargetSchemaName>).

Transformieren der XML-Datei

Der Ordner, in dem die XSLT-Datei gespeichert wird, enthält auch eine Batch-Datei mit dem Namen **DoTransform.bat**, die die XML-Datei mittels AltovaXML transformiert. AltovaXML kann kostenlos von der Altova Website heruntergeladen werden.

So transformieren Sie die persönliche Reisekostenabrechnung in die Firmen-Spesenabrechnung:

1. Laden Sie den kostenlosen AltovaXML-Prozessor von der [AltovaXML Download-Seite](#) herunter und installieren Sie ihn. AltovaXML wird standardmäßig im Verzeichnis c:\Program Files\Altova\AltovaXML\2012\ installiert.
2. Starten Sie die Batch-Datei **DoTransform.bat**, die sich im zuvor angegebenen Ausgabeordner befindet.
Daraufhin wird im aktuellen Ordner ...\\Tutorial die Ausgabedatei **ExpReport-Target.xml** generiert.

Anmerkung: Eventuell müssen Sie den AltovaXML Installationsordner zur Pfadvariablen der Umgebungsvariablen hinzufügen.

4.4 Arbeiten mit mehreren Zielschemas / -dokumenten

In diesem Abschnitt wird erklärt, wie man ein zweites Zielschema / -dokument erstellt, in das die Datensätze übernommen werden, die **nicht die Reisekosten selbst** betreffen. Wir arbeiten hierbei mit der Tutorial-Beispieldatei **Tut-ExpReport.mfd**, die wir bis jetzt benutzt haben.

Ziel

In diesem Abschnitt des Tutorials erfahren Sie, wie man eine zweite Zielkomponente hinzufügt und mehrere Ausgabeschemas generiert. Dabei werden folgende Schritte beschrieben:

- [Erstellen einer zweiten Schemazielkomponente](#)
- [Herausfiltern aller nicht reisekostenbezogenen Ausgaben](#) in Ihrem Beispielbericht
- [Definieren mehrerer Zielschemas mit demselben Namen](#)
- [Anzeigen einer bestimmten Ausgabe](#)
- [Generieren von XSLT-Code für mehrere Zielschemas](#)
- [Generieren von Programmcode für mehrere Zielschemas](#)

In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.

Eigenschaften: Befindet sich im Menü **Komponenten**. Mit diesem Befehl können Sie im Quell- und Zielschema unter anderem die Schemadatei, die XML-Input-Datei und die XML-Output-Datei definieren.



Vorschau: Wird in der Titelleiste von Komponenten angezeigt, wenn mehrere Zieldateien definiert wurden. Klicken Sie auf dieses Symbol, um eine bestimmte Komponente für die Ausgabevorschau auszuwählen.



Generierte Ausgabe speichern: Befindet sich im Menü/Fenster **Ausgabe**. Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld **Speichern unter** aufzurufen und den Ordner auszuwählen, in dem die generierten Ausgabedaten gespeichert werden sollen.



Ausgabe validieren: Befindet sich im Menü/Fenster **Ausgabe**. Klicken Sie auf diese Schaltfläche, um zu überprüfen, ob die generierte Ausgabe gültig ist. Das Ergebnis der Validierung wird im Fenster "Meldungen" angezeigt.

Datei | Code generieren in: Wählen Sie diese Option, um die Ausgabesprache (d.h. **XSLT 1.0** oder **XSLT 2.0**) auszuwählen. Mit dem Befehl wird das Dialogfeld **Verzeichnis auswählen** geöffnet, wo Sie den Ordner definieren, in dem der generierte XSLT-Code gespeichert werden soll.

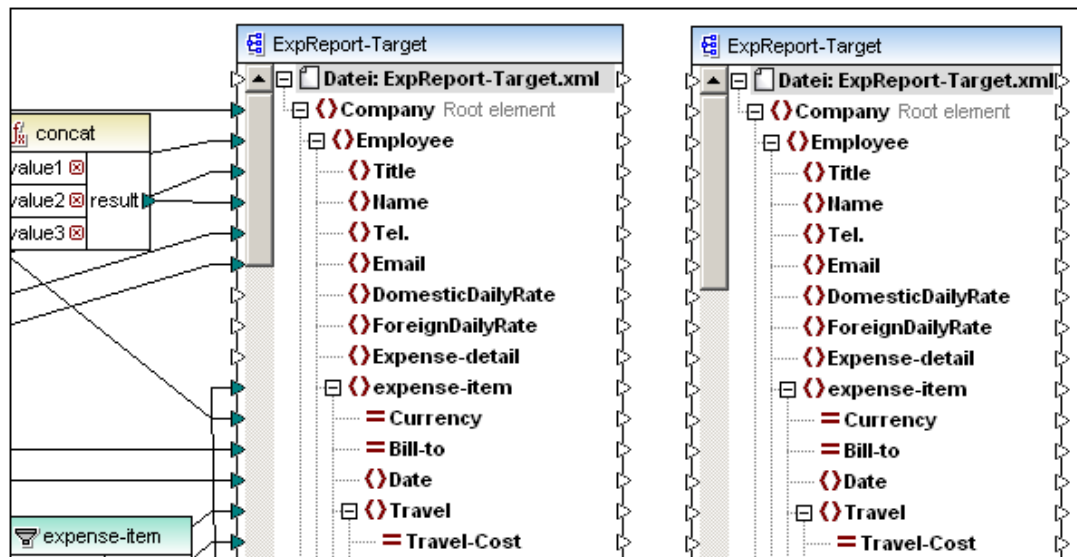
4.4.1 Erstellen einer zweiten Zielschemakomponente

In diesem Abschnitt des Tutorials wird erklärt, wie man eine zweite Zielschemakomponente erstellt, in der alle nicht reisekostenbezogenen Daten herausgefiltert werden.

Erzeugen einer zweiten Zielschemakomponente:

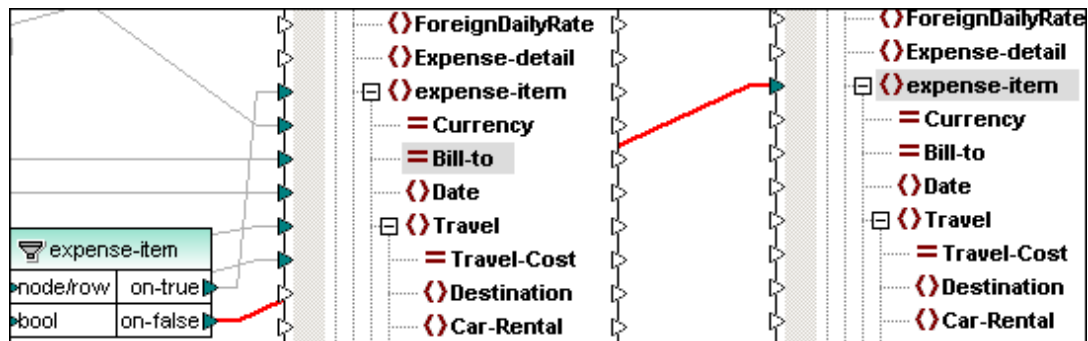
1. Klicken Sie auf das Symbol **XML-Schema / Datei einfügen**.
2. Wählen Sie im Dialogfeld "Öffnen" die Datei **ExpReport-Target.xsd**.
Sie werden nun aufgefordert, eine XML-Beispieldatei für dieses Schema anzugeben.
3. Klicken Sie auf **Überspringen** und wählen Sie **Company** als Root-Element des Zieldokuments aus.
Die Zielschemakomponente wird nun im Mapping-Fenster angezeigt.
4. Klicken Sie auf den Eintrag **Company** und drücken Sie auf der Zahlentastatur die *-Taste, um alle Datenelemente anzuzeigen.
5. Klicken Sie auf das Symbol zum Vergrößern des Fensters und passen Sie die Größe der Komponente an. Platzieren Sie die Schemakomponenten so, dass Sie gut zu sehen sind, damit sie optimal damit arbeiten können.

Wir haben nun ein Quellschema **mf-expReport** und zwei Zielschemas, beide mit dem Namen **ExpReport-Target**, die im Mapping-Fenster zu sehen sind.



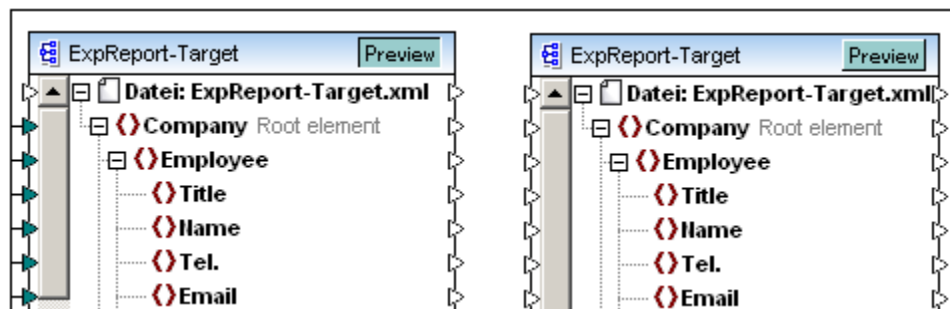
Herausfiltern von Daten, die nicht die Reiskosten selbst betreffen:

1. Verbinden Sie das **on-false**-Symbol der **filter**-Komponente mit dem Element **expense-item** des **zweiten** Zielschemas / -dokuments.



Daraufhin wird eine Meldung eingeblendet, dass Sie nun mit mehreren Zielschemas / -dokumenten arbeiten.

- Bestätigen Sie dies mit **OK**.



In der Titelleiste der einzelnen Zielschemakomponenten wird nun ein **Vorschau**symbol angezeigt.

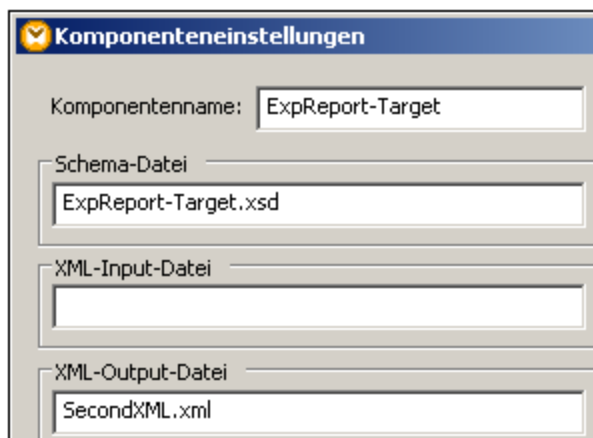
Durch Klicken auf dieses Symbol definieren Sie, welche der Zielschemadaten angezeigt werden sollen, wenn Sie anschließend auf die Schaltfläche XSLT, XSLT2, XQuery oder Ausgabe klicken.

Definition mehrerer Zielschemas desselben Namens für die Code-Generierung

In diesem Beispiel haben beide Zielschemas denselben Namen. Wir müssen daher sicherstellen, dass der generierte Code die erste Ergebnisdatei nicht mit der zweiten überschreibt. Beim Generieren von XSLT/XQuery ist dies nicht notwendig.

So definieren Sie die XML-Ausgabedatei:

- Rechtsklicken Sie auf das **zweite** Zielschema/ -dokument (ExpReport-Target) und wählen Sie die Option **Eigenschaften**.
- Geben Sie im Feld **XML-Output-Datei** einen Dateinamen ein, z.B. **...\\MapForceExamples\\Tutorial\\SecondXML.xml**.



3. Klicken Sie auf **OK** um das Dialogfeld **Eigenschaften** zu schließen.
Es wird empfohlen bei der Code-Generierung den **absoluten** Pfad einzufügen. Beim obigen Beispiel wird der Standardinstallationspfad von MapForce verwendet. Beachten Sie, dass der Name der Ausgabedatei in der Komponente nun **SecondXML.xml** lautet.

Erzeugen von Mappings für die übrigen Daten der Reisekostenabrechnung:

Erstellen Sie die folgenden Mappings **zwischen der Schemaquell- und Zielkomponente**. Sie haben die gleichen Konnektoren für das erste Zielschema erstellt, gehen Sie daher vor wie vorhin:

- Person auf Employee
- Tittle auf Title
- Phone auf Tel.
- Email auf Email
- currency auf Currency
- expto auf Bill-to
- Date auf Date

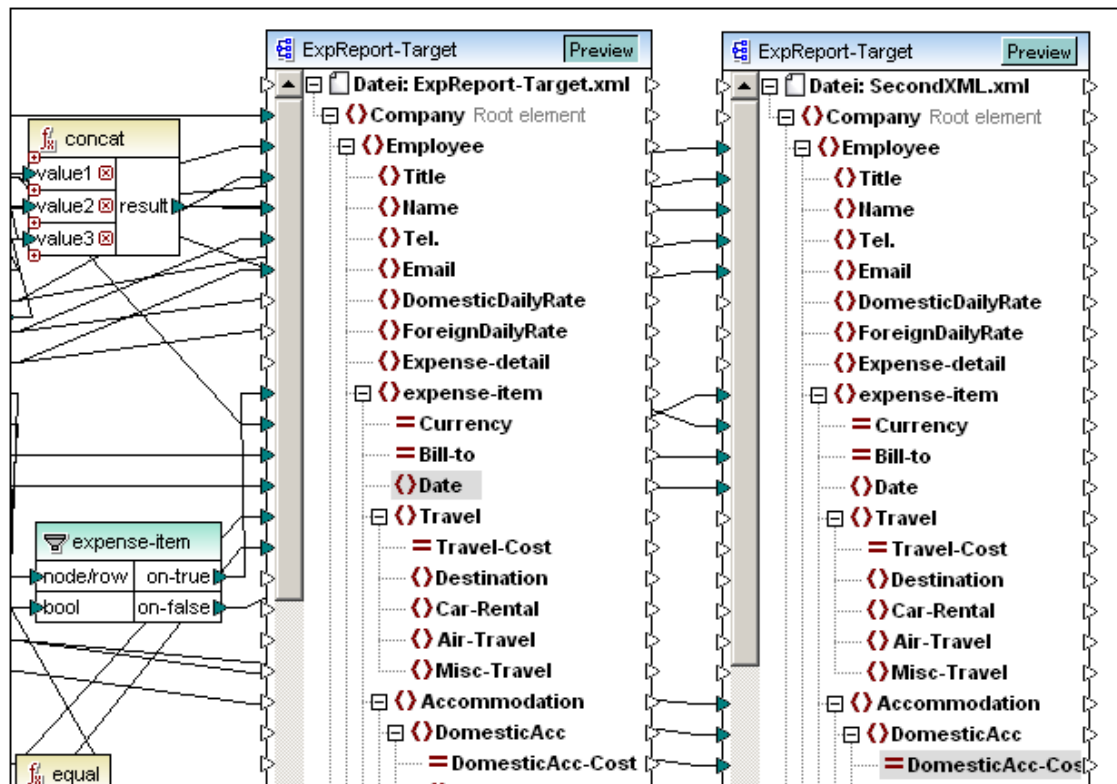
Erstellen Sie das folgende Mapping zwischen der vorhandenen **concat-Funktion** und dem **zweiten Zielschema**:

- result auf Name

So erstellen sie die restlichen nicht reisekostenbezogenen Mappings:

Stellen Sie sicher, dass die Option "Identische Sub-Einträge automatisch verbinden" deaktiviert ist und

1. Verbinden Sie das Datenelement **Lodging** im Quellschema mit **Accommodation** im zweiten Zielschema.
2. Verbinden Sie das Element **Lodging** mit **DomesticAcc**.
3. Verbinden Sie das Datenelement **Lodge-Cost** mit **DomesticAcc-Cost**.

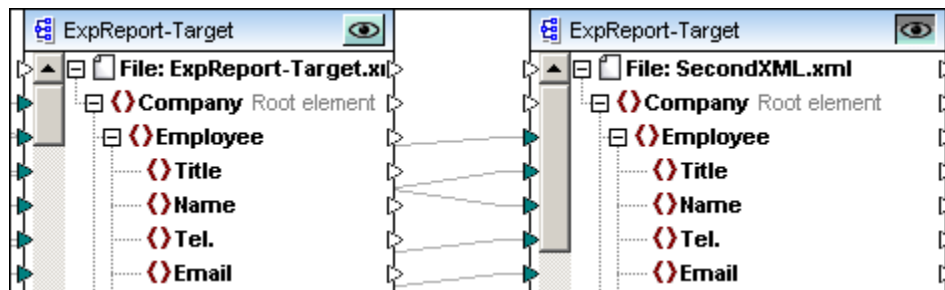


4.4.2 Anzeigen und Generieren mehrerer Zielschema-Ausgaben

Wenn Sie auf das Preview-Symbol klicken, können Sie auswählen, welches der Ziele angezeigt werden soll.

So zeigen Sie eine bestimmte XSLT-Ausgabe an:

1. Klicken Sie in der Titelleiste der **zweiten** Schemakomponente auf das **Vorschausymbol** , um es aktiv zu setzen (falls es noch nicht aktiv ist).



2. Klicken Sie im Mapping-Fenster auf die Schaltfläche **Ausgabe**.



Die XML-Ausgabe enthält zwei Datensätze, die beide für den Bereich Sales verrechnet werden: Die Kosten für "Domestic Accommodation" in der Höhe von \$121,2 und einen Expense-item-Datensatz, der nur ein Datum enthält. Dieser Datensatz stammt aus dem expense-item Meal. Gegenwärtig gibt es kein Mapping zwischen Meal costs und Domestic accommodation und selbst wenn eines vorhanden wäre, würden keine Kosten aufscheinen, da die XML-Instanz keine enthält.

Bitte beachten Sie: Sie können diese XML-Daten speichern. Klicken Sie dazu im Vorschau-Fenster für die Anzeige der XML-Ausgabe auf das Symbol **Ausgabedatei speichern**



Sie können die erzeugte XML-Instanzdatei auch gegen das Zielschema validieren, wenn Sie auf die Schaltfläche "Validieren" klicken .

So generieren Sie XSLT 1.0/XSLT 2.0-Code für mehrere Zielschemas:

1. Wählen Sie die Menüoption **Datei | Code generieren in | XSLT oder XSLT2..**

2. Wählen Sie den Ordner, in dem die erzeugten XSLT-Dateien gespeichert werden sollen, und klicken Sie auf OK.
Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
3. Navigieren Sie zum gewünschten Ordner, in dem sich nun zwei XSLT-Dateien mit den Namen **MappingExpReport-Target.xslt** und **MappingExpReport-Target2.xslt** befinden.

So transformieren Sie den persönlichen Ausgabebericht in den Firmen-Ausgabenbericht:

1. Laden Sie den kostenlosen AltovaXML Prozessor von der [AltovaXML-Download-Seite](#) herunter und installieren Sie ihn. AltovaXML wird standardmäßig im Ordner c:\Program Files\Altova\AltovaXML2012\ installiert.
2. Starten Sie die Batch-Datei **DoTransform.bat**, die sich im zuvor angegebenen Ausgabeordner befindet.
Daraufhin wird im Ordner ...\\Tutorial die Ausgabedatei **ExpReport-Target.xml** generiert.

Hinweis: Eventuell müssen Sie die Installationspfad von AltovaXML zur Pfadvariablen der Umgebungsvariablen hinzufügen.

So generieren Sie Programmcode für mehrere Zielschemas:

1. Wählen Sie die Menüoption **Datei | Code generieren in | XQuery, Java, C# oder C++**.
2. Wählen Sie den Ordner, in dem die erzeugten Dateien gespeichert werden sollen und klicken Sie auf OK.
Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
3. Navigieren Sie zum gewünschten Ordner und kompilieren Sie Ihr Projekt.
4. Kompilieren Sie den Programmcode und führen Sie ihn mit Ihrem Compiler aus.
Die Anwendung erzeugt zwei XML-Dateien.

Bitte beachten Sie: MapForce erstellt für die Kompilierung des [Java-Codes](#) eine **JBuilder**-Projektdatei und ein **Ant** Build Script. Nähere Informationen dazu finden Sie im Abschnitt [JDBC-Treiberinstallation](#).

4.5 Mappen mehrerer Quellelemente auf ein einziges Zielelement

In diesem Abschnitt werden wir zwei einfache Mitarbeiter-Reisekostenabrechnungen auf eine einzige Firmen-Spesenabrechnung mappen. Bei diesem Beispiel handelt es sich um eine vereinfachte Version des Mappings, das Sie im Tutorial bereits im Abschnitt über das [Mappen mehrerer Zielschemas / - dokumente](#) durchgeführt haben.

Bitte beachten Sie: Es gibt eine alternative Methode dazu, dies mit der Funktion für [dynamischen Input/Output](#) von Komponenten zu tun. Ein Beispiel dazu finden Sie unter "[Dynamische Dateinamen - Input / Output](#)".

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man zwei **persönlichen Reisekostenabrechnungen** in einer Firmen-Spesenabrechnung zusammenführt. Dabei werden folgende Schritte beschrieben:

- [Mappen von Schemakomponenten](#) (Wiederholung)
- [Duplizieren von Input-Datenelementen](#)
- [Entfernen von duplizierten Datenelementen](#)

In diesem Abschnitt verwendete Befehle



Neu...: Dieser Befehl befindet sich im Menü **Datei** und in der Standard-Symbolleiste. Klicken Sie auf diese Schaltfläche, um das Dialogfeld **Neue Datei** aufzurufen, in dem Sie ein neues Mapping erstellen können.



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identischer Sub-Einträge ein- bzw. auszuschalten.



Duplik davor/danach einfügen: Befindet sich im Kontextmenü, das angezeigt wird, wenn Sie mit der rechten Maustaste auf ein Datenelement in einer Komponente klicken. Klicken Sie auf diesen Befehl, um das ausgewählte Datenelement zu duplizieren.



Duplikat löschen: Befindet sich im Kontextmenü, das angezeigt wird, wenn Sie mit der rechten Maustaste auf ein Datenelement in einer Komponente klicken. Klicken Sie auf diesen Befehl, um das ausgewählte Datenelement aus der Komponente zu löschen.

In diesem Abschnitt verwendete Dateien

Die für dieses Beispiel verwendeten Dateien wurden speziell adaptiert, um zu demonstrieren, wie Daten aus zwei XML-Eingabedateien auf ein einziges Zieldatenelement im Zielschema gemappt werden können. Dieses Beispiel ist kein reales Szenario.

mf-ExpReport.xml

mf-ExpReport2.xml

mf-ExpReport-combined.xml

ExpReport-combined.xsd

die im vorigen Abschnitt verwendete

XML-Eingabedatei

die zweite XML-Eingabedatei

die nach erfolgtem Mapping erzeugte Datei

die Zielschemadatei, in der die beiden XML-Quelldateien zusammengeführt werden sollen

Tut-ExpReport-msource.mfd

die Mapping-Datei für dieses Beispiel

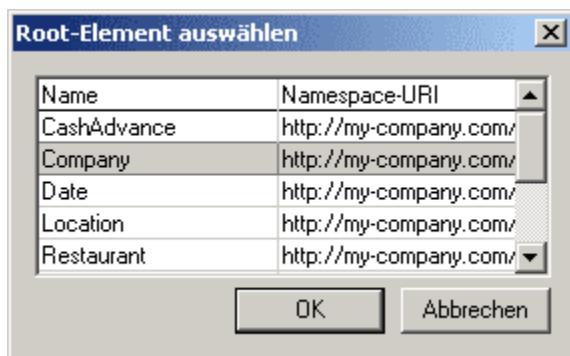
Bitte beachten Sie: Die in diesem Abschnitt verwendeten Dateien liegen ebenfalls im Ordner [...\MapForceExamples\Tutorial\](#).

4.5.1 Erzeugen der Mappings

In der unten stehenden Anleitung wird rekapituliert, wie man eine Mapping-Umgebung einrichtet. Dieses Mapping finden Sie im Ordner [...\\MapForceExamples\\Tutorial\\](#) unter dem Namen **Tut-ExpReport-msource.mfd**.

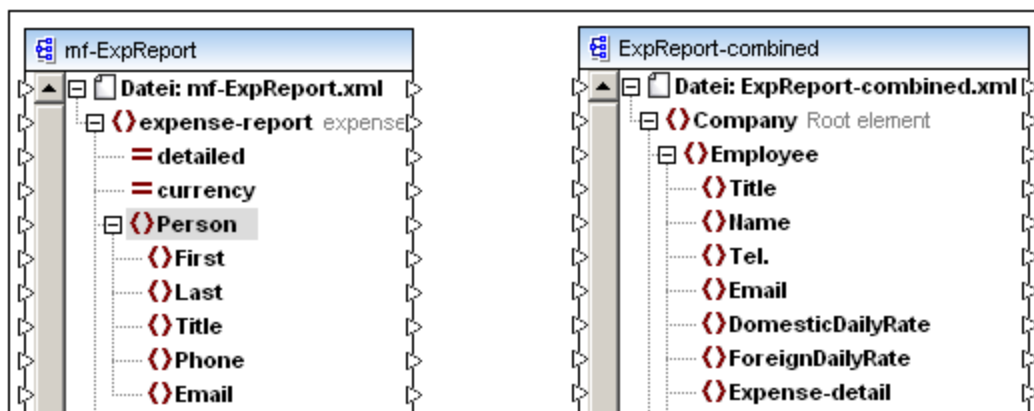
So erstellen Sie die Mapping-Umgebung:

1. Klicken Sie in der Standard-Symbolleiste auf die Schaltfläche **Neu**  um das Dialogfeld **Neue Datei** zu öffnen.
2. Klicken Sie auf das Mapping-Symbol und anschließend auf **OK**, um ein neues Mapping-Register aufzurufen.
3. Klicken Sie auf die Schaltfläche **XML-Schema/Datei einfügen** .
4. Wählen Sie im Dialogfeld "Öffnen" die Datei **mf-ExpReport.xsd** aus dem Unterordner "Tutorial" im Verzeichnis MapForceExamples, klicken Sie auf **Durchsuchen...** und wählen Sie die Datei **mf-ExpReport.xml** als XML-Instanzdatei, wenn Sie danach gefragt werden.
5. Klicken Sie auf den Eintrag **expense-report**, drücken Sie auf der Zahlentastatur die *-Taste, um alle Datenelemente einzublenden; passen Sie ggf. die Größe der Komponente an.
6. Klicken Sie auf das Symbol **XML-Schema / Datei einfügen** .
7. Wählen Sie im Dialogfeld "Öffnen" die Datei **ExpReport-combined.xsd**. Sie werden nun aufgefordert, eine XML-Beispieldatei für dieses Schema anzugeben.
8. Klicken Sie auf "Überspringen" und wählen Sie **Company** als Root-Element des Zieldokuments.



Die Zielschema-Komponente wird nun im Mapping-Fenster angezeigt.

9. Klicken Sie auf den Eintrag **Company**, drücken Sie die *-Taste, um alle Elemente einzublenden und passen Sie ggf. die Fenstergröße an.

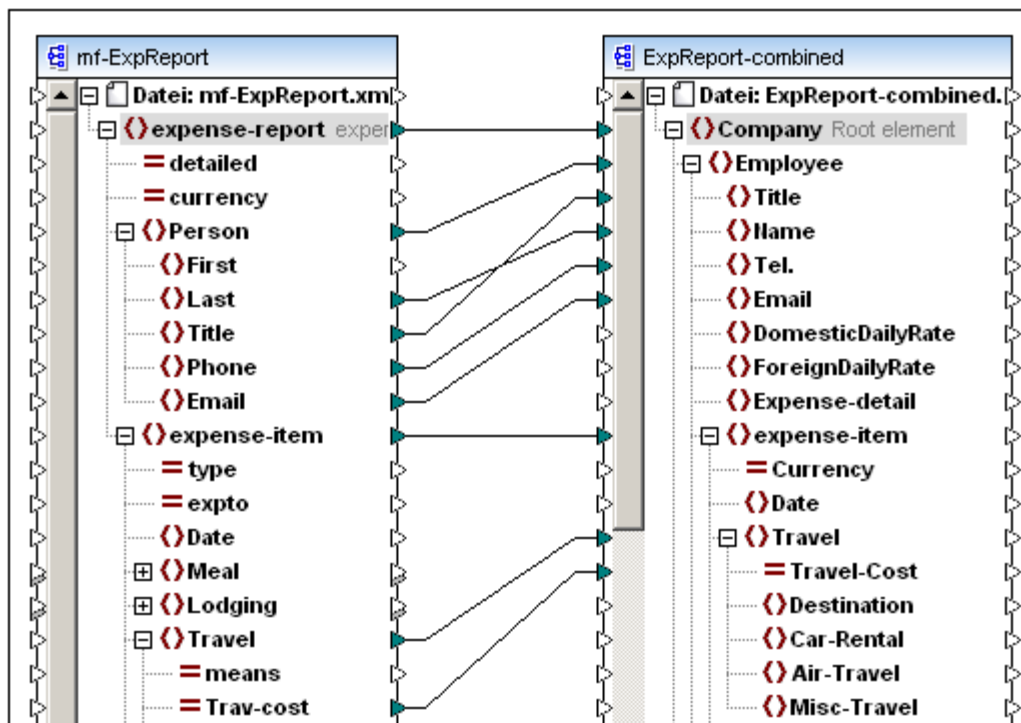


Mappen der Komponenten

Vergewissern Sie sich, dass das Symbol "Identische Sub-Einträge automatisch verbinden"  deaktiviert ist, bevor Sie die folgenden Mappings erzeugen.

- Expense-report auf Company
- Person auf Employee
- Last auf Name
- Title auf Title
- Phone auf Tel.
- Email auf Email
- expense-item auf expense-item
- Travel auf Travel und
- Trav-cost auf Travel-Cost

Das Mapping wird unten stehend angezeigt.



Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des aktuellen Mappings

anzuzeigen.

```
1      <?xml version="1.0" encoding="UTF-8"?>
2      <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.w3.org/2001/XMLSchema-instance http://www.w3.org/2001/XMLSchema-instance">
3          <Employee>
4              <Title>Project Manager</Title>
5              <Name>Landis</Name>
6              <Tel.>123-456-78</Tel.>
7              <Email>f.landis@nanonull.com</Email>
8              <expense-item>
9                  <Travel Travel-Cost="337.88"/>
10             </expense-item>
11             <expense-item/>
12             <expense-item>
13                 <Travel Travel-Cost="1014.22"/>
14             </expense-item>
15             <expense-item>
16                 <Travel Travel-Cost="2000"/>
17             </expense-item>
18             <expense-item/>
19         </Employee>
20     </Company>
21
```

Bitte beachten Sie: Wenn ein **gemapptes Parent-Element** in der Quelldatei Child-Elemente enthält, die nicht auf das Zielschema gemappt wurden, werden leere <expense-item/> Elemente/Tags generiert. In diesem Fall wurden nur die Travel-Elemente des übergeordneten Elements expense-item gemappt. Die Liste enthält jedoch noch zwei weitere expense-Elemente: ein lodging-Element und ein meal-Element. Für jedes dieser Elemente wird ein leerer übergeordneter expense-item Tag erzeugt.

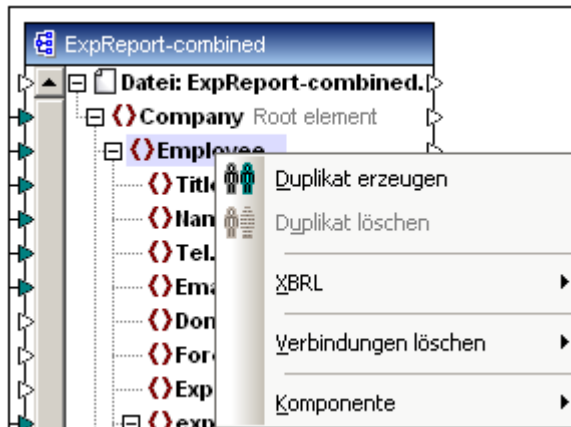
Um keine leeren Tags zu generieren, erstellen Sie einen Filter, wie zuvor im Tutorial unter "[Filtern von Daten](#)" beschrieben, oder verbinden Sie das Datenelement **Travel** mit dem Datenelement **expense-item**.

4.5.2 Duplizieren von Input-Datenelementen

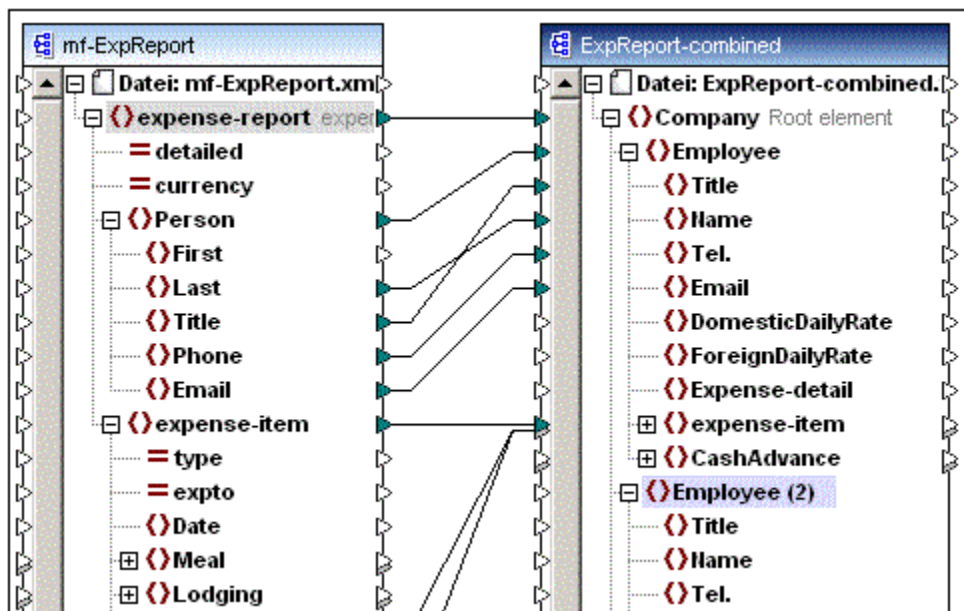
Um mehrere Quelldatenelemente auf ein und dasselbe Zieldatenelement mappen zu können, müssen wir nun die **Eingabelemente** der Zielkomponente duplizieren, um Mappings aus einer anderen XML-Quelldatei zu erzeugen. Zu diesem Zweck fügen wir die zweite XML-Quelldatei hinzu und erstellen ein Mapping von dieser zu den gleichen Inputs des duplizierten Datenelements in der XML-Zieldatei.

Duplizieren von Datenelementen:

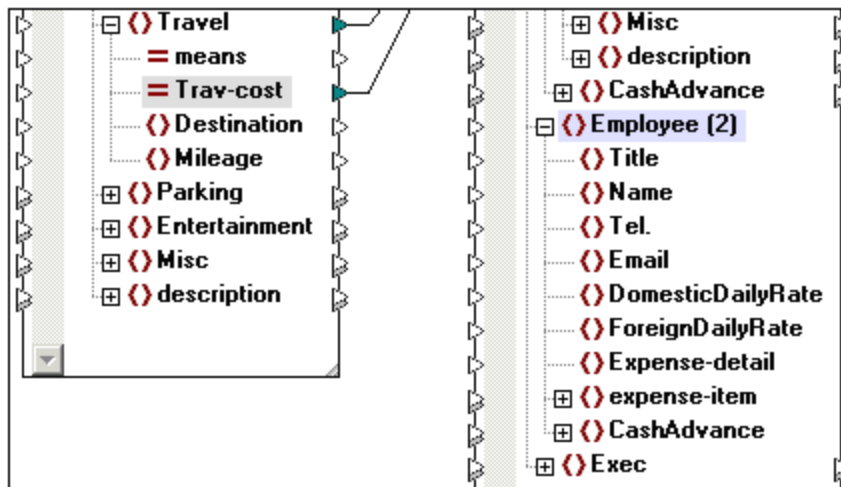
1. Rechtsklicken Sie in der XML-Zieldatei auf den Eintrag Employee.
2. Wählen Sie im Kontextmenü die Option **Duplikat danach einfügen**.



Der Komponente wurde nun ein zweites Element Employee hinzugefügt, nämlich **Employee(2)**.



3. Klicken Sie auf das Erweiterungssymbol, um die untergeordneten Elemente anzuzeigen.
Die **Struktur** des neuen Employee-Elements ist eine exakte Kopie des Originals mit der einzigen Ausnahme, dass die duplizierten Elemente über **kein Output-Symbol** verfügen.



Sie können diese neuen duplizierten Elemente nun als **Zielelemente** für die zweite XML-Quelldatei verwenden.

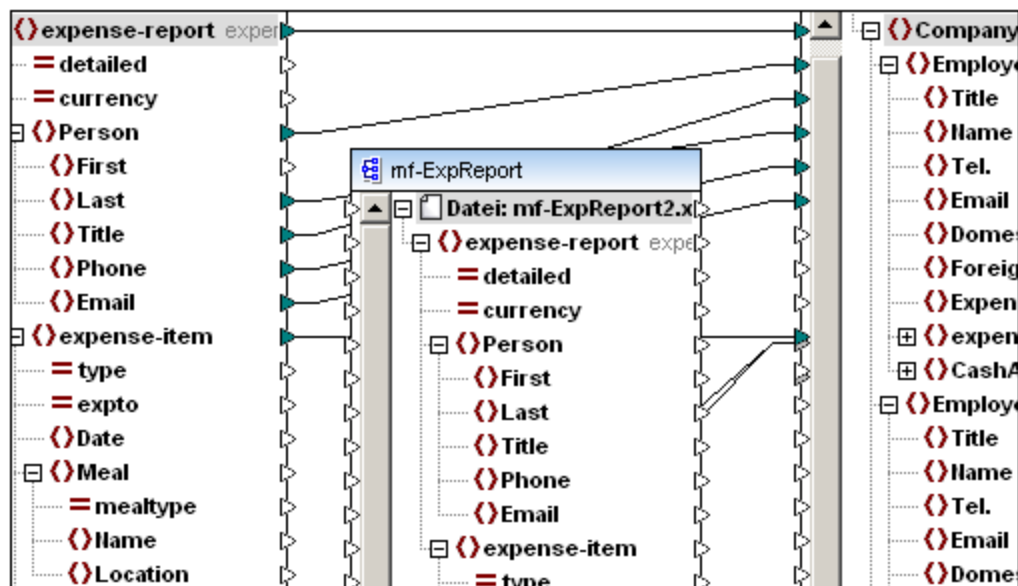
Einfügen der zweiten XML-Instanzdatei

Um die zweite XML-Instanzdatei einzufügen, gehen Sie vor wie oben beschrieben und verwenden Sie dazu dieselbe XML-Schemadatei wie zuvor.

So fügen Sie eine zweite Quellkomponente ein:

1. Klicken Sie auf das Symbol **XML-Schema/Datei einfügen** .
2. Wählen Sie im Dialogfeld "Öffnen" die Datei **mf-ExpReport.xsd**, klicken Sie auf "Durchsuchen..." und wählen Sie als XML-Instanzdatei die Datei **mf-ExpReport2.xml**.
3. Klicken Sie auf den Eintrag **expense-report**, drücken Sie auf der Zahlentastatur die *-Taste, um alle Elemente einzublenden und passen Sie die Größe der Komponente ggf. an.

Aus Gründen der Übersichtlichkeit wurde die neue Komponente in der folgenden Abbildung zwischen den beiden bestehenden platziert.

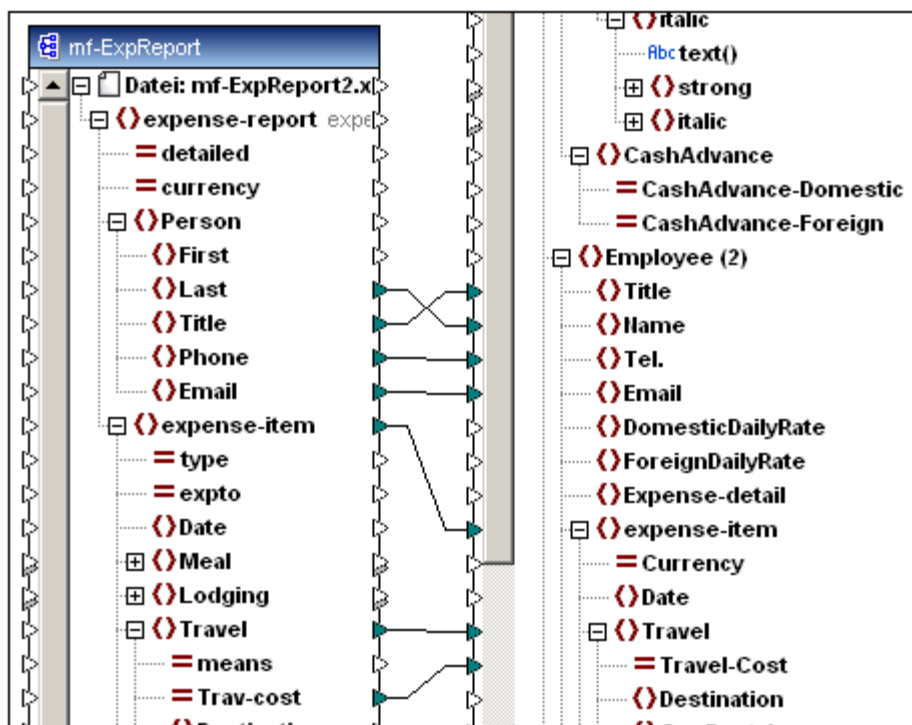


4. Erzeugen Sie dieselben Mappings wie für die erste XML-Quelldatei:

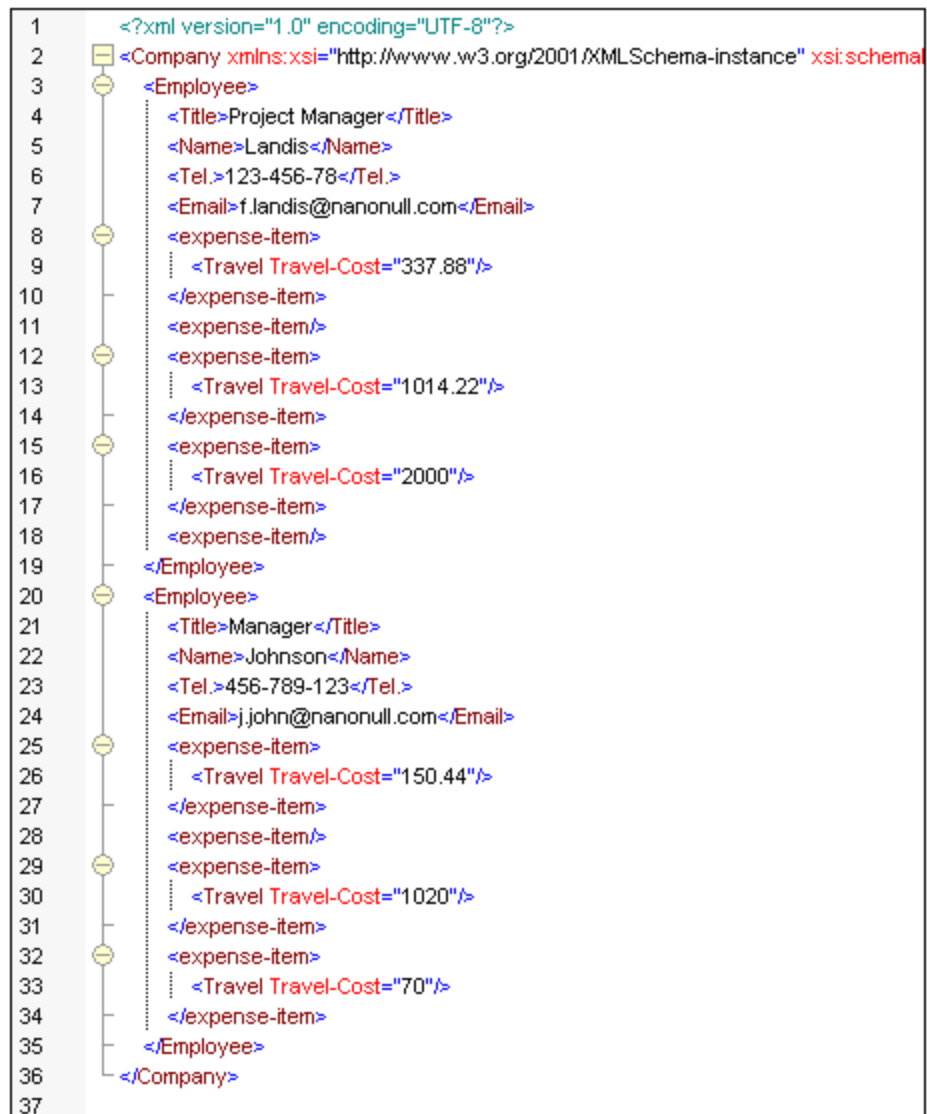
- Person auf Employee(2)
- Last auf Name
- Title auf Title
- Phone auf Tel.
- Email auf Email
- expense-item auf expense-item

Führen Sie einen Bildlauf nach unten durch und mappen Sie

- Travel auf Travel und
- Trav-cost auf Travel-Cost



5. Klicken Sie auf die Schaltfläche "Ausgabe", um das Mapping-Ergebnis anzuzeigen.



Die Daten der zweiten Reisekostenabrechnung wurden der Ausgabedatei hinzugefügt. Johnson und seine Reisekosten wurden in der Firmen-Spesenabrechnung zu den Ausgaben-Elementen von Fred Landis hinzugefügt.

So speichern Sie die generierte Ausgabe in einer Datei:

Klicken Sie auf das Symbol **Generierte Ausgabe speichern** , das Sie in der Titelleiste sehen, wenn das Fenster "Ausgabe" aktiv ist.

Sie finden die Datei mf-ExpReport-combined.xml im Ordner [...\MapForceExamples\Tutorial\](#).

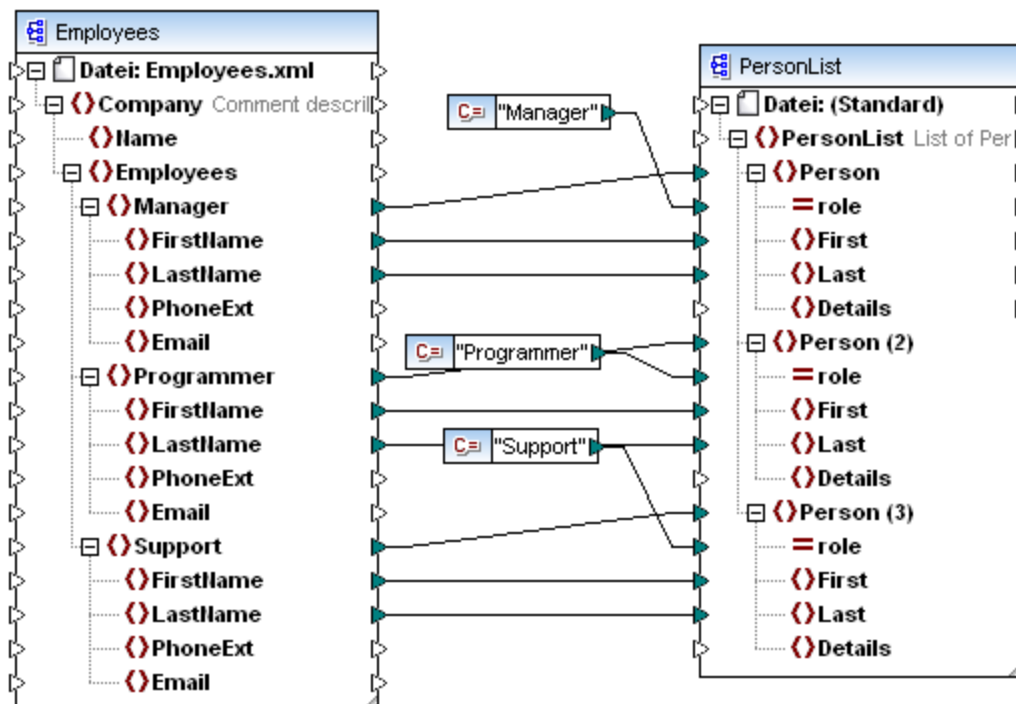
So löschen Sie duplizierte Datenelemente:

- Rechtsklicken Sie auf das duplizierte Datenelement und wählen Sie im Menü die Option **Duplikat löschen**.

Beispiel

Ein weiteres Beispiel, bei dem duplizierte Datenelemente verwendet werden, finden Sie in der Beispieldatei **PersonList.mfd** im Ordner [...\MapForceExamples\](#).

Im Beispiel **PersonList.mfd** werden unterschiedliche Datenelemente im Quelldokument auf "dasselbe" Element im Schema/XML-Zieldokument gemappt und spezifische Elemente (Manager usw.) mit Hilfe eines "role"-Attributs auf ein einziges generisches gemappt.



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <PersonList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="
   C:/AltovaMapForce2011/MapForceExamples/PersonList.xsd">
3    <Person role="Manager">
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6    </Person>
7    <Person role="Programmer">
8      <First>Frank</First>
9      <Last>Further</Last>
10   </Person>
11   <Person role="Support">
12     <First>Loby</First>
13     <Last>Matise</Last>
14   </Person>
15   <Person role="Support">
16     <First>Susi</First>
17     <Last>Sanna</Last>
18   </Person>
19 </PersonList>

```

4.6 Input aus mehreren Dateien / Output in mehrere Dateien

In diesem Abschnitt werden die neuen MapForce Funktionen für den Komponenten-Input aus mehreren Dateien / für den Komponenten-Output in mehrere Dateien beschrieben. Beachten Sie bitte, dass diese Funktionalität für XSLT 1.0 und XQuery nicht zur Verfügung steht.

Eine einzige Input-Komponente verarbeitet zwei Quelldokumente, während eine einzige Output-Komponente zwei Ausgabedateien generiert. Das hier verwendete Beispiel wurde im Abschnitt [Filtern von Daten](#) eingerichtet und dient auch als Basis im Abschnitt [Mappen mehrere Quellelemente auf ein einziges Zielelement](#).

Ziel

Ziel dieses Abschnitts ist die Erstellung eines Mappings, in dem die Quellkomponente zwei XML-Input-Dateien verarbeitet und die Zielkomponente zwei XML-Zieldateien ausgibt.

In diesem Abschnitt verwendete Befehle



Alle Ausgabedateien speichern...: Befindet sich im Menü **Ausgabe**. Wählen Sie diesen Befehl um alle gemappten Dateien aus dem Vorschaufenster zu speichern.

In diesem Abschnitt verwendete Beispieldateien:

mf-ExpReport.xml

Im vorigen Abschnitt verwendete
XML-Input-Datei

mf-ExpReport2.xml

die zweite XML-Input-Datei

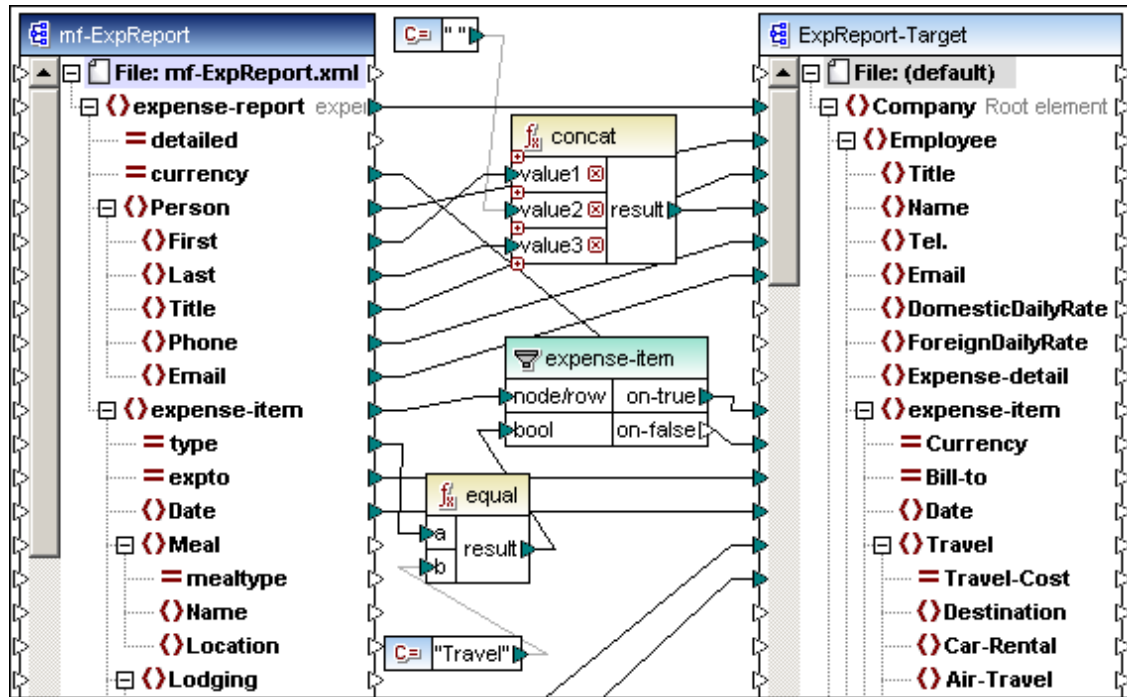
Tut-ExpReport-multi.mfd

Die Mapping-Datei für dieses Beispiel

Bitte beachten Sie: Die in diesem Abschnitt verwendeten Dateien stehen auch im Ordner [...\MapForceExamples\Tutorial\](#) zur Verfügung.

4.6.1 Verarbeiten mehrerer Dateien pro Input-/Output-Komponente

Die Datei **Tut-ExpReport.mfd** im Ordner MapForceExamples wird in diesem Beispiel geändert und unter einem anderen Namen gespeichert.



Beachten Sie bitte die folgenden Datenelemente am oberen Rand jeder Komponente:

- Das Datenelement **Datei:mf-ExpReport.xml** von mf-ExpReport enthält den Input/Output-XML-Dateieintrag. Wenn die Input- und die Output-Datei identisch sind, wird ein Eintrag angezeigt; wenn dies nicht der Fall ist, wird **Input-Dateiname;Output-Dateiname** angezeigt.

Dies wird automatisch ausgefüllt, wenn Sie einer XML-Schemadatei eine XML-Instanzdatei zuweisen.

- Das Datenelement **Datei: (Standard)** von ExpReport-Target zeigt an, dass der XML-Schemakomponente beim Einfügen keine Instanzdatei zugewiesen wurde., d.h. das Output-XML-Dateifeld ist leer. Daher wird bei der Ausführung des Mappings ein Standardwert verwendet.

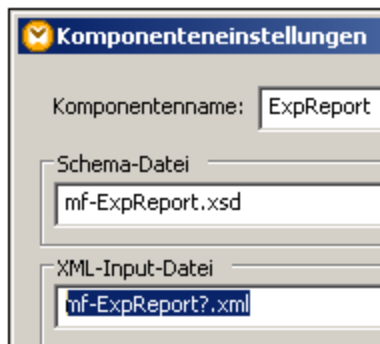
Verarbeiten mehrerer Dateien

Um mehrere Dateien verarbeiten zu können, verwendet MapForce im Dateinamen der XML-Input-Datei den Platzhalter "?". Das Zeichen "?" kann durch kein Zeichen oder ein Zeichen ersetzt werden.

So werden mehrere Dateien verarbeitet:

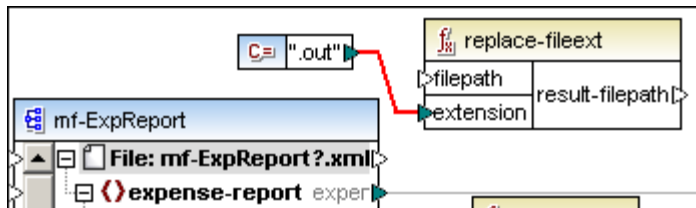
Geöffnet ist die Datei "Tut-ExpReport" aus dem Ordner ...Tutorial, wobei Sie als Ausgabesprache in der Symbolleiste auf "XSLT 2.0" geklickt haben:

1. Doppelklicken Sie auf der linken Seite auf die Komponente **mf-ExpReport**.
2. Fügen Sie mf-expReport?.xml in das Feld "XML-Input-Datei" ein.

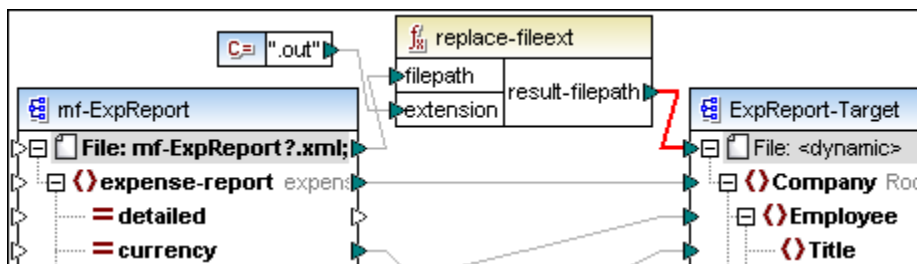


In Dateinamen werden die Platzhalterzeichen ? und * unterstützt. Beachten Sie, dass hier ein relativer Pfad eingegeben wurde, da die Datei Tut-ExpReport.mfd sich im Ordner ...\\Tutorial befindet (Auf Wunsch kann auch ein absoluter Pfad verwendet werden).

3. Fügen Sie die Funktion **replace-fileext** aus der Bibliothek **file path functions** ein und fügen Sie anschließend eine Konstantenkomponente ein.
4. Fügen Sie in die Konstantenkomponente ".out" ein und verbinden Sie sie mit dem **extension**-Parameter der Funktion.



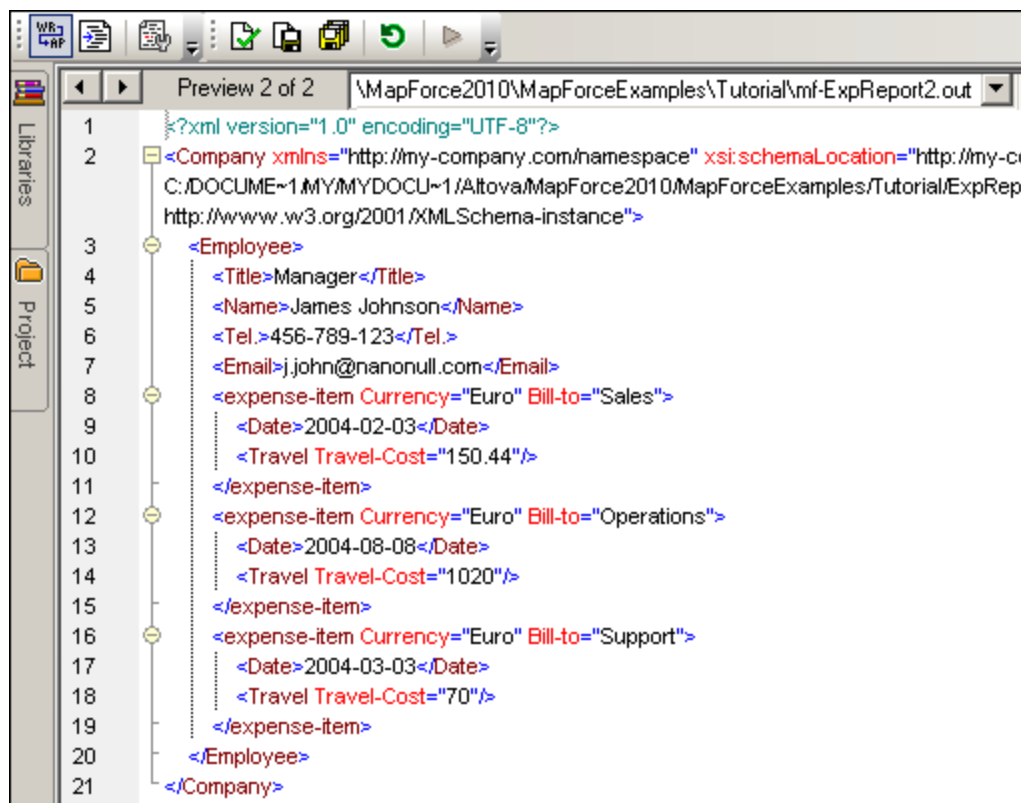
5. Verbinden Sie das Datenelement **Datei:mf-ExpReport?.xml** der Komponente mit dem **filepath**-Parameter der Funktion.
6. Verbinden Sie den **result-filepath**-Parameter der Funktion mit dem Datenelement **Datei <Standard>** der Zielkomponente.



Das Datenelement File: der Zielkomponente hat sich nun ebenfalls in **Datei: <dynamisch>** geändert.

7. Klicken Sie auf die Schaltfläche "Ausgabe", um die Ergebnisse zu sehen. Im Ausgabefenster sehen Sie jetzt im Vorschaufenster die Ergebnisse für jede XML-Input-Datei, z.B. Vorschau 1 von 2, wie unten in der Abbildung gezeigt.

8. Klicken Sie auf den Bildlaufpfeil, um das Ergebnis der zweiten XML-Input-Datei zu sehen.
Beachten Sie, dass in der Auswahlliste die Namen der einzelnen XML-Quelldateien angezeigt werden, wobei die Erweiterung ***.xml** durch die Erweiterung ***.out** ersetzt wurde.



Wenn Sie auf die Schaltfläche **Alle speichern**  klicken, werden alle gemappten Dateien aus dem Vorschaufenster gespeichert, ohne dass Code generiert werden muss. Es erscheint eine Meldung, wenn die Ausgabedateien im selben Ordner überschrieben werden.

9. Speichern Sie die Mapping-Datei unter einem neuen Namen.

Bitte beachten Sie: Nähere Informationen zu mehreren Input-/Output-Dateien finden Sie unter [Dynamischer Input/Output](#).

4.7 Mappen einer Datenbank auf ein Schema

In diesem Abschnitt wird gezeigt, wie man eine einfache Microsoft Access-Datenbank als Datenquelle verwendet, um Datenbankdaten auf ein Schema zu mappen.

Ziel

In diesem Abschnitt lernen Sie, wie man eine Datenbank in ein MapForce-Mapping einfügt und die Tabellen dieser Datenbank auf die Schemazielkomponente mappt. Es werden die folgenden Schritte beschrieben:

- [Herstellen einer Verbindung zu einer Microsoft Access-Datenbank](#)
- [Mappen von Datenbanktabellen auf XML-Schema-Nodes](#)

In diesem Abschnitt verwendete Befehle



Neu...: Dieser Befehl befindet sich im Menü **Datei** und in der Standard-Symbolleiste. Klicken Sie auf diese Schaltfläche, um das Dialogfeld **Neue Datei** aufzurufen, in dem Sie ein neues Mapping erstellen können.



Datenbank einfügen: Klicken Sie auf diese Schaltfläche, um das Dialogfeld **Datenbank auswählen** aufzurufen, in dem Sie eine Verbindung zu einer Datenbank herstellen können.



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identer Sub-Einträge ein- bzw. auszuschalten.

Code generieren in: Befindet sich im Menü **Datei**. Wenn Sie diesen Befehl auswählen, wird ein Untermenü geöffnet, in dem Sie die gewünschte Art von Code (d.h. **Java**, **C++** oder **C#**) auswählen können. Mit dem Befehl wird das Dialogfeld **Verzeichnis auswählen** geöffnet, in dem Sie definieren können, wo der generierte Code gespeichert werden soll.

Derzeit werden die folgenden Datenbanken (und Verbindungsarten) unterstützt:

- Microsoft Access; Version 2003 / 2007
- Microsoft SQL Server Versionen 2000, 2005 und 2008
- Oracle Versionen 9i, 10g und 11g
- MySQL 4.x und 5.x
- PostgreSQL 8.0, 8.1, 8.2, 8.3
- Sybase 12
- IBM DB2 Version 8.x und 9
- IBM DB2 für i 5.4
- IBM DB2 für i 6.1

Bitte beachten Sie:

MapForce 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.

Stellen Sie bei Installation der **64-Bit** Version von **MapForce** sicher, dass Sie Zugriff auf die 64-Bit-Datenbanktreiber haben, die für die jeweilige Datenbank, zu der Sie eine Verbindung herstellen möchten, benötigt werden.

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.

In der unten stehenden Tabelle sehen Sie die erzeugten Datenbanktypen, die Einschränkungen und die Verbindungsmethoden für das Einbinden der Datenbanken.

	Datenbank einfügen - Verbindungsmethoden ("Schema" anhand einer Datenbank erzeugen)	
Unterstützte Datenbank	ODBC-Einschränkungen (Eindeutige Schlüssel werden von ODBC nicht unterstützt)	ADO-Einschränkungen
Microsoft Access (ADO)	OK Primär- und Sekundärschlüssel werden nicht unterstützt	OK *
MS SQL Server (ADO)	OK	OK *
Oracle (ODBC)	OK, Einschränkungen: Tabellen mit Spalten vom Typ CLOB, BLOB, BFILE; XML-Tabellen	OK, Einschränkungen: Tabellen mit Spalten vom Typ CLOB, BLOB, BFILE; XML-Tabellen; Owner-Informationen, Identity Constraints werden aus der Datenbank nicht ausgelesen
MySQL (ODBC)	OK *	OK †
Sybase (ODBC)	OK *	OK
IBM DB2 (ODBC)	OK *	OK

* **Empfohlene Verbindungsmethode für die einzelnen Datenbanken.**

† **MySQL: Beim Herstellen der ADO-Verbindung auf Basis von ODBC wird empfohlen, entweder die Benutzer- oder die System-DSN zu verwenden.**

- **Nicht verfügbar**

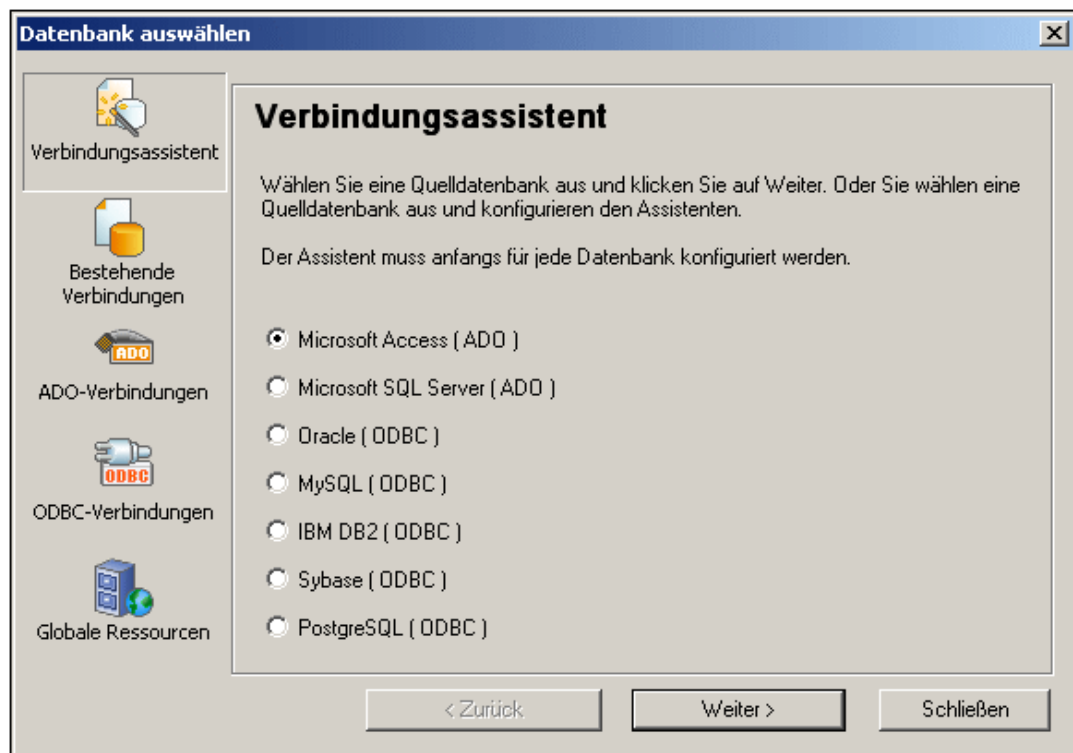
Bitte beachten Sie:

Datenbanken können auch als "globale Ressourcen" eingefügt werden. Nähere Informationen dazu finden Sie unter [Globale Ressourcen](#).

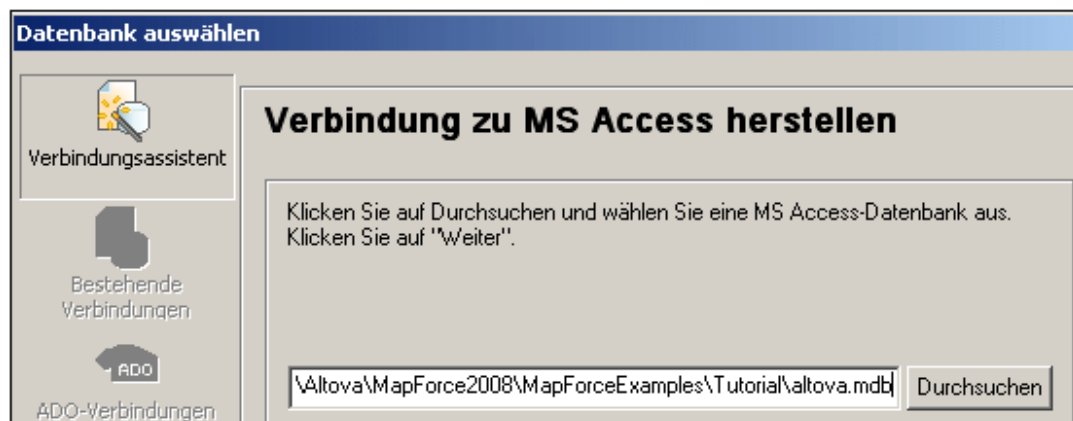
4.7.1 Herstellen einer Verbindung zu einer Access-Datenbank

Erzeugen der Datenbankkomponente in MapForce:

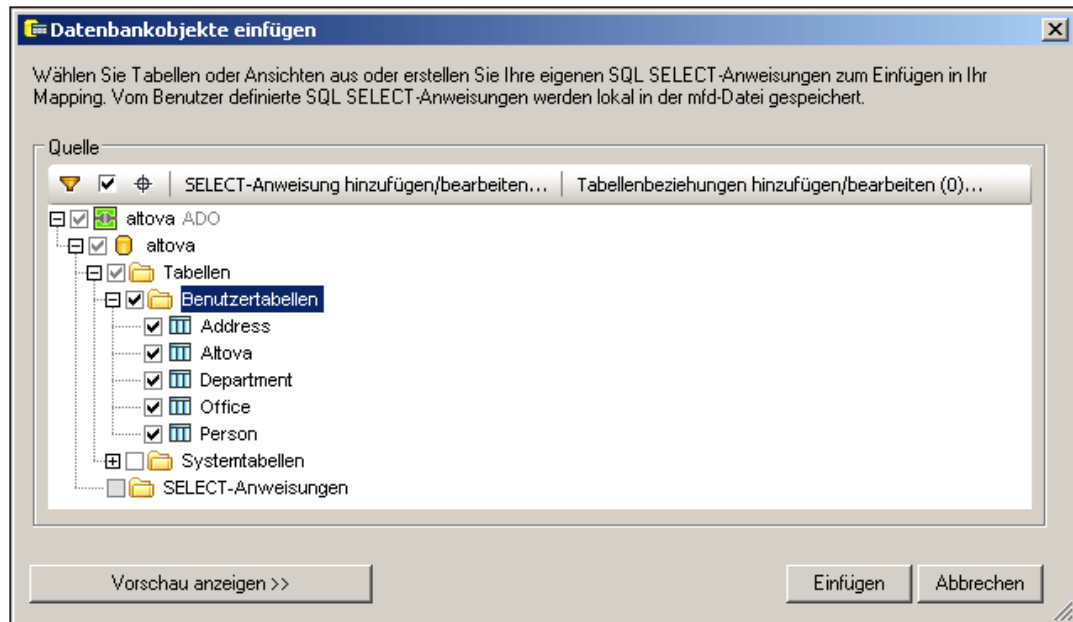
1. Wählen Sie in MapForce **Datei | Neu**, um ein neues Mapping zu erstellen.
2. Klicken Sie in der Titelleiste auf eine der Schaltflächen : Java, C#, C++ oder BUILTIN.
3. Klicken Sie in der Symbolleiste auf das Symbol **Datenbank einfügen**.
4. Klicken Sie im Dialogfeld **Datenbank auswählen** auf die Schaltfläche **Verbindungsassistent**.



4. Klicken Sie auf das Optionsfeld **Microsoft Access**.
5. Klicken Sie zum Fortfahren auf "Weiter".
6. Klicken Sie auf die Schaltfläche "Durchsuchen", um die gewünschte Datenbank als Datenquelle auszuwählen: In diesem Fall ist die Quelle **altova.mdb** im Ordner [...\\MapForceExamples\\Tutorial\\](#).
Im Textfeld wird der Connection String angezeigt.



7. Klicken Sie auf **Verbinden**.



8. Klicken Sie im Dialogfeld **Datenbankobjekte einfügen** auf das Kontrollkästchen links von den Benutzertabellen und anschließend auf **Einfügen** um die Datenbank-(Schema-) Komponente einzufügen. Damit werden alle Tabellen des Ordners ausgewählt.



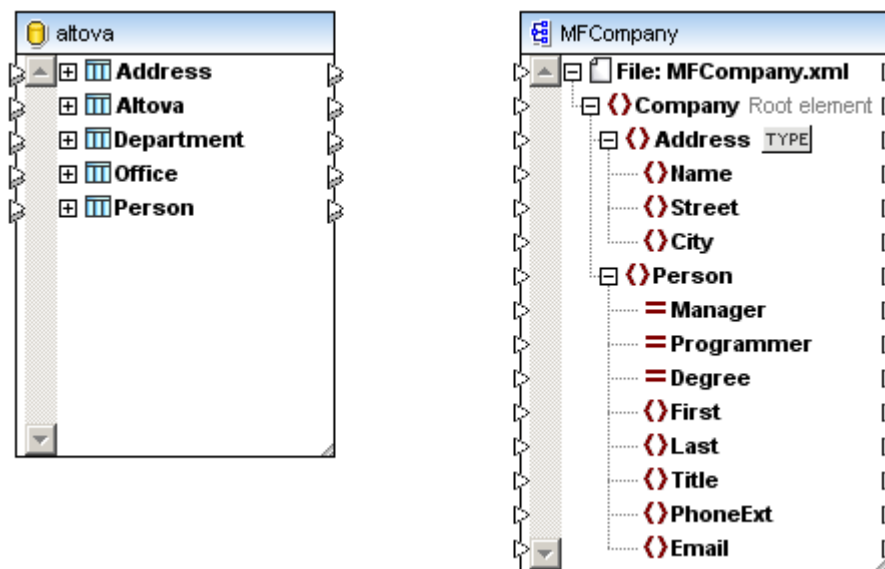
Die Datenbankkomponente wird im Mapping-Fenster angezeigt. Sie können nun Mappings auf ein Zielschema/XML-Dokument erzeugen.

4.7.2 Mappen von Datenbankdaten

Nachdem Sie die Datenbankkomponente nun in das Mapping-Fenster eingefügt haben, können Sie die Zielschemakomponente hinzufügen und die Mappings zwischen den Komponenten herstellen.

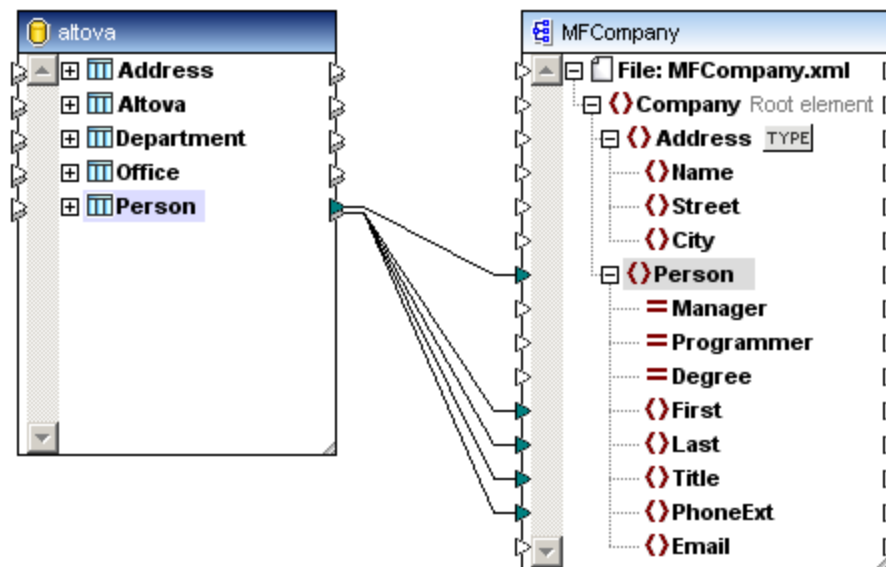
Einfügen des Zielschemas / -dokuments:

1. Klicken Sie auf das Symbol **XML-Schema / Datei einfügen** und wählen Sie das Schema **MFCCompany.xsd** aus.
2. Klicken Sie auf **Überspringen**, wenn Sie aufgefordert werden, eine Beispieldatei anzugeben.
3. Wählen Sie **Company** als Root-Element und erweitern Sie alle Datenelemente. Sie können nun mit dem Mappen der Datenbankdaten auf das Schema / XML-Dokument beginnen.



Mappen von Datenbankdaten auf ein Schema/Dokument in MapForce

1. Aktivieren Sie das Symbol "Identische Sub-Einträge automatisch verbinden" , wenn es noch nicht aktiv ist.
2. Klicken Sie in der Datenbankkomponente auf das "Tabellenelement" **Person** und verbinden Sie es mit dem Element **Person** in MFCCompany. Daraufhin werden für alle Datenelemente desselben Namens in den beiden Komponenten Konnektoren erzeugt.



4. Speichern Sie die MapForce-Datei, z.B. als **PersonDB**.
5. Klicken Sie auf die Schaltfläche Ausgabe, um das Ergebnis/eine Vorschau des Mappings anzuzeigen. Die MapForce Engine generiert Ergebnisse on-the-fly, ohne dass Sie dafür Code generieren oder kompilieren müssen.

Generieren von Java-Code und der XML-Ergebnisdatei:

1. Wählen Sie die Menüoption **Datei | Code generieren in | Java**.
2. Wählen Sie das Verzeichnis aus, in dem die Java-Dateien gespeichert werden sollen, und klicken Sie auf OK.
Nach erfolgreichem Abschluss des Vorgangs erscheint die Meldung "Codegenerierung wurde erfolgreich ausgeführt".
3. Kompilieren Sie den generierten Code und führen Sie ihn aus.
Es wird die folgende Datei MFCompany.xml erstellt.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3    <Person>
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6      <Title>Office Manager</Title>
7      <PhoneExt>582</PhoneExt>
8    </Person>
9    <Person>
10     <First>Frank</First>
11     <Last>Further</Last>
12     <Title>Accounts Receivable</Title>
13     <PhoneExt>471</PhoneExt>
14   </Person>
15   <Person>
16     <First>Loby</First>
17     <Last>Matise</Last>
18     <Title>Accounting Manager</Title>
19     <PhoneExt>963</PhoneExt>
20   </Person>

```

Kompliziertere Beispiele für das Mapping von Datenbankdaten auf Schemas unter Verwendung von:

- mehreren Quelldateien
- flachen und hierarchischen Datenbanken

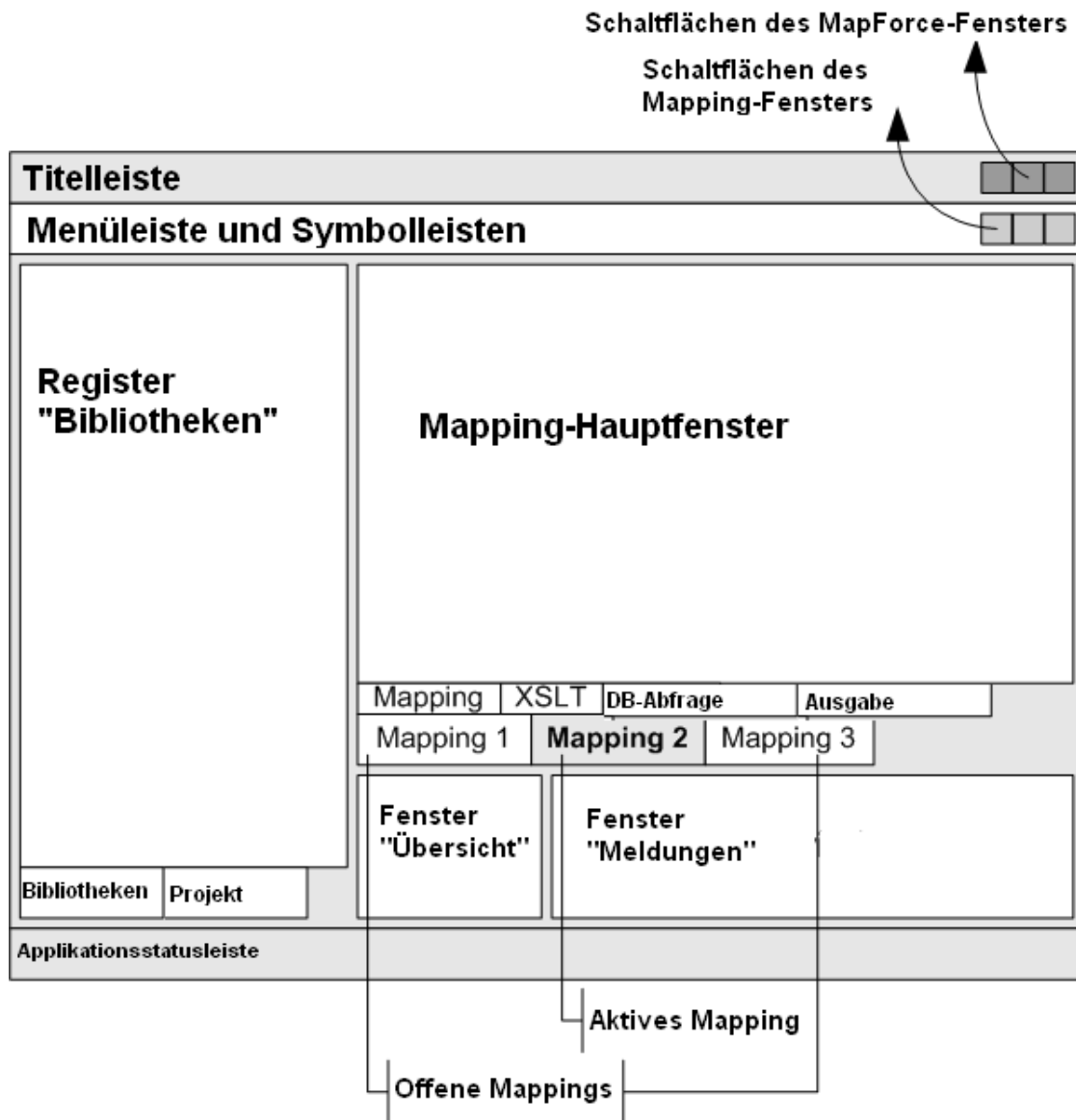
finden Sie in den Dateien **DB_AltovaSQLXML.mfd** und **DB_Altova_Hierarchical.mfd** im Ordner [...MapForceExamples](#) von MapForce>.

Kapitel 5

MapForce Benutzeroberfläche

5 MapForce Benutzeroberfläche

MapForce besteht aus vier Hauptbereichen: Dem Bereich [Bibliotheken](#) und [Projekt](#) auf der linken Seite, dem Bereich "Mapping" (mit den Fenstern [Mapping](#), [XSLT](#), [XSLT2](#), [XQuery](#), [DBQuery](#), und [Ausgabe](#)) auf der rechten Seite sowie den Fenstern [Übersicht](#) und [Meldungen](#) unterhalb davon.



Titelleiste

In der Titelleiste werden der Name der Applikation (d.h. MapForce) gefolgt vom Namen des aktiven Mapping Design-Fensters angezeigt. Auf der rechten Seite befinden sich Schaltflächen zum Steuern des MapForce-Applikationsfensters.

Menüleiste und Symbolleisten

In der Menüleiste werden die Menübefehle angezeigt. Jede Symbolleiste enthält eine Gruppe von Symbolen, die für MapForce-Befehle stehen. Sie können die Menüleiste und die Symbolleisten an ihren Ziehpunkten mit der Maus an die gewünschte Stelle ziehen.

Register "Bibliotheken" und "Projekt"

Das Register [Bibliotheken](#) enthält je nach der gewählten Ausgabesprache unterschiedliche Funktionen. Sie können eine Funktion direkt in das Mapping-Fenster ziehen.

Das Register [Projekt](#) zeigt alle Dateien und Ordner an, die für das jeweilige Projekt bzw. den jeweiligen Webservice relevant sind. Sie können Ordner und Dateien zum Projekt hinzufügen und über das Menü **Projekt** Standardprojekteinstellungen definieren.

Mapping-Fenster

Im Fenster [Mapping](#) werden die grafischen Elemente dargestellt, die zur Erstellung des [Mappings \(der Transformation\) zwischen den verschiedenen Komponenten](#) verwendet werden. Im Quellschema wird die Struktur des Quellschemas angezeigt und im Zielschema wird die Struktur des Zielschemas angezeigt. Die Input- und Output-**Symbole** der einzelnen Schema-Datenelemente werden durch **Konnektoren** verbunden. Bei Schema-**Datenelementen** kann es sich entweder um Elemente oder Attribute handeln.

Die folgenden Fenster können durch Klicken auf die entsprechenden Schaltflächen am unteren Rand des Mapping-Fensters angezeigt werden:

- In den Fenstern **XSLT**, **XSLT2** und **XQuery** wird je nachdem, [welche Sprache ausgewählt ist](#), eine Vorschau des Transformationscodes angezeigt.
- Im Fenster **DB Query** können Sie alle gängigen [Datenbanken](#) direkt abfragen. Die hier definierten Abfragen/Aktionen sind unabhängig von allen anderen MapForce-Registern und werden nicht im Rahmen der *.MFD-Datei gespeichert.

Die einzelnen Datenbankabfragefenster sind mit dem aktuell aktiven Mapping verknüpft, sodass pro Sitzung/Mapping **mehrere Datenbankabfragen** ausgeführt werden können. Bitte beachten Sie, dass Sie für jedes Datenbankabfragefenster auch mehrere aktive Verbindungen zu unterschiedlichen Datenbanken haben können.
- Im Fenster **Ausgabe** wird eine Vorschau der transformierten oder gemappten Daten in einer Textansicht angezeigt.
- In den Fenstern **HTML, RTF, PDF und Word 2007+** werden die Daten der Zielkomponente als HTML-, RTF-, PDF- oder Word 2007+-Dokumente angezeigt, wenn ein [StyleVision Power Stylesheet \(SPS\) mit der Zielkomponente verknüpft ist](#).

Das Fenster "Datenbankabfrage"

Wenn Sie auf die Schaltfläche **DB-Abfrage** klicken, werden weitere Fenster, Symbolleisten und eine Statusleiste angezeigt, über die Sie eine Verbindung zu einer Datenbank herstellen, Datenbanken anzeigen und mit diesen arbeiten können (*Abbildung unten*).



Über das Fenster **DB-Abfrage** können Sie jede gängige [Datenbank](#) direkt abfragen. Die hier definierten Abfragen/Aktionen sind unabhängig von allen anderen MapForce-Registern und werden nicht im Rahmen der *.MFD-Datei gespeichert.

Das Fenster "DB-Abfrage" ist in die folgenden Bereiche gegliedert:

- Im [Online Browser](#) werden Verbindungsinformationen und Datenbanktabellen angezeigt sowie eine komplette Übersicht über die Objekte in den einzelnen Datenbanken, u.a. Datenbank-Constraint-Informationen, wie z.B. eine Spalte als Primär- oder Sekundärschlüssel. In IBM DB2 Version 9-Datenbanken werden im Browser zusätzlich dazu noch registrierte XML-Schemas in einem separaten Ordner angezeigt.
- Mit dem [SQL Editor](#) werden SQL-Anweisungen geschrieben und ausgeführt. Er enthält auch Funktionen wie z.B. [automatische Generierung](#) und [automatische Komplettierung](#) von SQL-Anweisungen, Definition von [Regionen](#) oder Einfügen von [Zeilen- und Blockkommentaren](#).

- Das Ergebnis- und Meldungsfenster enthält das Register [Ergebnisse](#), auf dem die Abfrageergebnisse in Tabellenform angezeigt werden und das Register [Meldungen](#), in dem Warnungen und Fehlermeldungen angezeigt werden.

Jedes Datenbankabfragefenster ist mit dem aktuell aktiven Mapping verknüpft, sodass **mehrere Datenbankabfragen** pro Sitzung/Mapping möglich sind. Beachten Sie, dass für jedes Datenbankabfrageregister gleichzeitig auch mehrere Verbindungen zu unterschiedlichen Datenbanken aktiv sein können.

Datenbankabfrage-Statusleiste

In der Datenbankabfrage-Statusleiste am unteren Rand des Ergebnisfensters werden Informationen über den Fortschritt der Abfrage angezeigt: ob die Daten erfolgreich abgerufen wurden oder der Vorgang abgebrochen oder vom Benutzer gestoppt wurde. Außerdem werden die Anzahl der abgerufenen Zeilen und Spalten sowie der Zeitpunkt und die Dauer der Ausführung der Datenbankabfrage angezeigt.

Fenster "Übersicht" und "Meldungen"

Im Fenster **Übersicht** wird der Mappingbereich mit einem roten Rechteck markiert. Sie können das Rechteck verschieben, um in Ihrem Mapping zu navigieren.

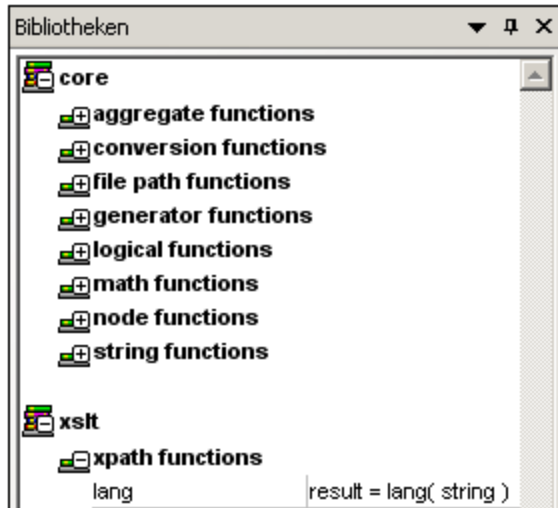
Im Fenster **Meldungen** werden alle Validierungswarnungen oder Fehlermeldungen angezeigt, die während eines Mappings auftreten können. Wenn Sie auf eine Meldung in diesem Fenster klicken, wird sie auf dem Mapping-Register markiert, damit Sie den Fehler beheben können.

Applikationsstatusleiste

Die Statusleiste der Applikation befindet sich am unteren Rand des Applikationsfensters. In ihr werden Informationen auf Applikationsebene angezeigt, darunter auch Tooltips, die eingeblendet werden, wenn Sie den Mauszeiger über eine Symbolleisten-Schaltfläche platzieren. Die Applikationsstatusleiste ist nicht mit der Datenbankabfragestatusleiste zu verwechseln. Wenn Sie die 64-Bit-Version von MapForce verwenden, wird in der Statusleiste das Suffix (x64) hinter dem Applikationsnamen angezeigt. Für die 32-Bit-Version gibt es kein Suffix.

5.1 Das Register "Bibliotheken"

Im **Bibliotheksfenster** werden die für die aktuell ausgewählte Programmiersprache verfügbaren Bibliotheken sowie deren einzelne **Funktionen** angezeigt. Sie können Funktionen mit der Maus direkt in das **Mapping-Fenster** ziehen, wo sie zu Funktionskomponenten werden.



XSLT ausgewählt

Die Standardbibliotheken **core**, **lang**, **xpath2**, **edifact**, **xbri**, **xsix** und **xslt** werden bei jedem Start von MapForce automatisch geladen und müssen nicht mehr vom Benutzer hinzugefügt werden. Die Bibliothek **Core** enthält eine Sammlung von Funktionen, mit Hilfe derer alle Arten von Ausgabecode wie XSLT, XQuery, Java, C# und C++ generiert werden können. In den anderen Bibliotheken (xslt, xslt2, xpath2, lang usw.) sind Funktionen zusammengefasst, die sich auf die einzelnen Ausgabearten beziehen.

Bitte beachten Sie: Die XPath 2.0 Bibliothek und ihre Funktionen stehen sowohl für XSLT 2.0 als auch XQuery zur Verfügung. Wenn Sie die Schaltfläche für den

Built-In-Ausführungsprozessor  aktiviert haben, stehen dieselben Funktionen auch für Java, C# und C++ zur Verfügung.

Bei Auswahl von:

XSLT werden die Funktionen der Core- und der XSLT-Bibliothek (XPath 1.0 und XSLT 1.0-Funktionen) aktiviert.

XSLT2 werden die Funktionen der Core-, XPath 2.0 und XSLT 2.0-Bibliothek aktiviert.

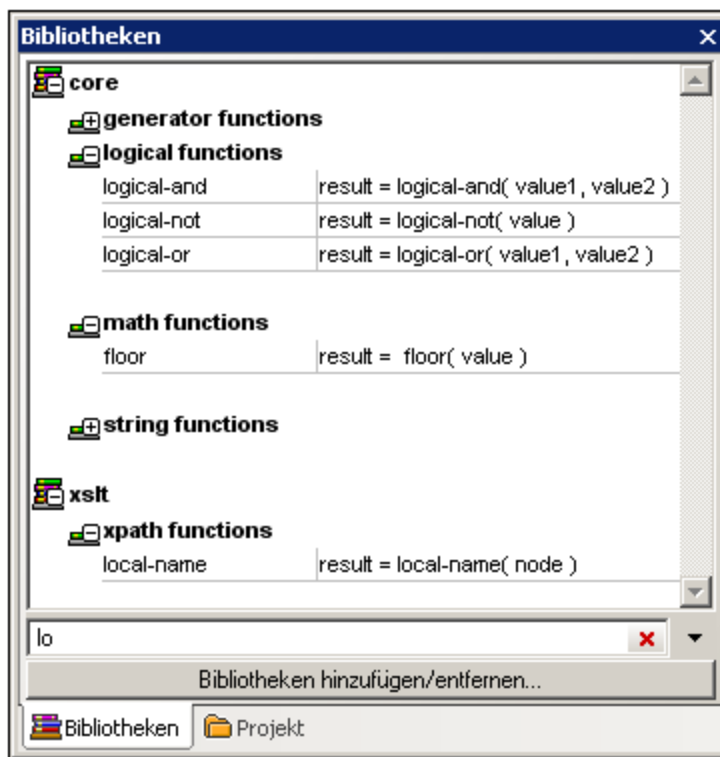
XQ(uey) werden die Funktionen der Core- und der XPath 2.0-Bibliothek aktiviert.

Einschränkungen bei XPath 2.0:

Einige XPath 2.0-Funktionen für die Verarbeitung von Sequenzen sind derzeit nicht verfügbar.

Suchen von Funktionen im Fenster "Bibliotheken"

Am unteren Rand des Registers "Bibliotheken" befindet sich ein Suchfeld, über das Sie nach Funktionsnamen suchen können.

*XSLT ausgewählt*

Wenn Sie auf die **Esc**-Taste drücken, wird der Filter im Fenster gelöscht. Ein Klick auf **x** hat denselben Effekt.

So suchen Sie im Fenster "Bibliotheken" nach einer Funktion:

1. Klicken Sie in das Fenster "Bibliotheken" um es aktiv zu machen und geben Sie die gesuchten Zeichen ein, z.B. "lo".

Alle Funktionen, die diese Zeichen enthalten, werden nun im Fenster "Bibliotheken" in der jeweiligen Funktionsgruppe angezeigt.

3. Klicken Sie auf den Abwärtspfeil und wählen Sie "Funktionsbeschreibungen inkludieren", wenn der Text der Funktionsbeschreibungen in der Funktionssuche berücksichtigt werden soll.



Hinzufügen neuer Funktionsbibliotheken

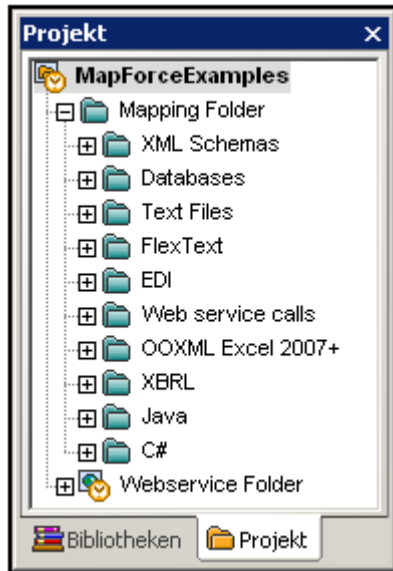
In MapForce können Sie Ihre eigenen Funktionsbibliotheken erstellen und integrieren. Nähere Informationen dazu finden Sie unter "[Hinzufügen benutzerdefinierter Funktionsbibliotheken](#)", "[Benutzerdefinierte XSLT 1.0-Funktionen](#)", "[Benutzerdefinierte XSLT 2.0-Funktionen](#)" und "[Benutzerdefinierte Funktionen](#)".

Bitte beachten Sie: Benutzerdefinierte Funktionen/Bibliotheken können auch für Java, C# und C++ sowie XSLT und XSLT 2 definiert werden.

5.2 Das Register "Projekt"

MapForce unterstützt die Mehrfachdokumentenschnittstelle (MDI) und gestattet Ihnen, Ihre Mappings in Mapping-Projekte zu gruppieren. Projektdateien haben die Erweiterung ***.mfp**. Auf dem Register "Projekt" werden alle Dateien und Ordner angezeigt, die zum Projektverzeichnis hinzugefügt wurden.

Anmerkung: Durch Drücken der Tastenkombination Strg + F können Sie auf dem Register "Projekt" nach Mappings suchen.



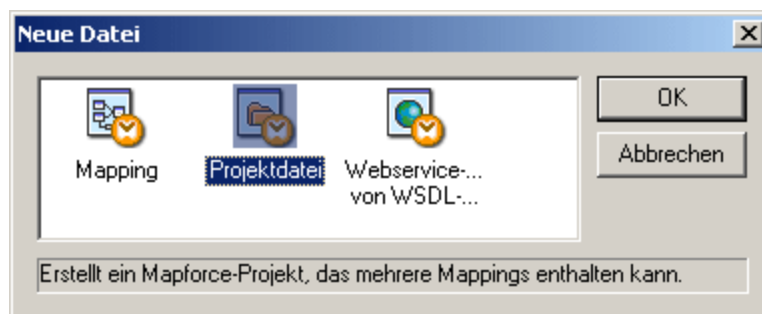
Es können zwei Arten von Projekten definiert werden:

- eine Sammlung einzelner Mappings, also ein Standardprojekt.
- eine Gruppe miteinander in Zusammenhang stehender Mappings, aus denen ein WSDL-Mapping-Projekt besteht.

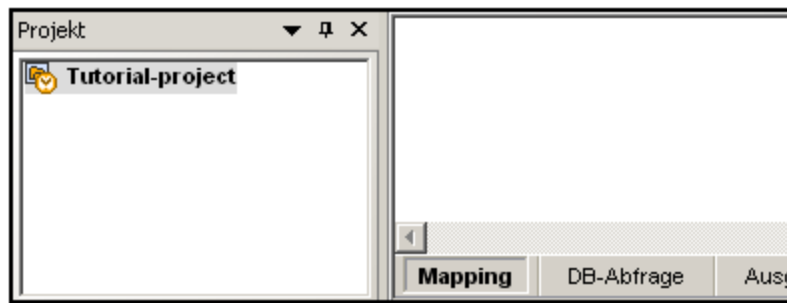
Beide Arten unterstützen die Codegenerierung für das gesamte Projekt.

So erstellen Sie ein neues Projekt:

1. Wählen Sie **Datei | Neu** und doppelklicken Sie auf das **Projektdatei**-Symbol.



2. Geben Sie den Namen des Projekts im Dialogfeld **Projekt speichern unter** ein und klicken Sie zum Fortfahren auf **Speichern**. Auf dem Register "Projekt" wird ein Projektordner hinzugefügt.



3. Wählen Sie **Datei | Neu** und doppelklicken Sie auf das Mapping-Symbol. Daraufhin wird im Mapping-Fenster eine neue Mapping-Datei mit dem Namen "Neues Design1" geöffnet.

Erstellen von Webservice-Projekten

Ein Webservice-Projekt unterscheidet sich insofern von einem Standardprojekt, als zur Erstellung eines solchen Projekts eine WSDL-Datei erforderlich ist und das Ergebnis der Codegenerierung ein vollständiger Webservice ist. Sie müssen nur noch den generierten Code kompilieren und den Webservice auf Ihrem jeweiligen Webserver bereitstellen.

Jede in der WSDL-Datei definierte Operation wird als einzelnes Mapping dargestellt. Nähere Informationen dazu finden Sie im Abschnitt [Implementieren von Webservices](#).

So erstellen Sie ein WSDL-Projekt:

1. Wählen Sie **Datei | Neu** und doppelklicken Sie auf das Symbol **Webservice-Projekt**.
2. Klicken Sie auf die Schaltfläche "Durchsuchen", um die Web Service Definition-Datei (*.wsdl) auszuwählen. Die restlichen Felder werden automatisch ausgefüllt.

Projektoptionen

Sie können Dateien zum Projekt hinzufügen, nachdem es erstellt wurde und eine Ordnerstruktur innerhalb des Projekts erstellen. Die entsprechenden Befehle stehen im Menü **Projekt** oder im Kontextmenü zur Verfügung, das angezeigt wird, wenn Sie mit der rechten Maustaste auf dem Register "Projekt" auf den Projektnamen klicken.



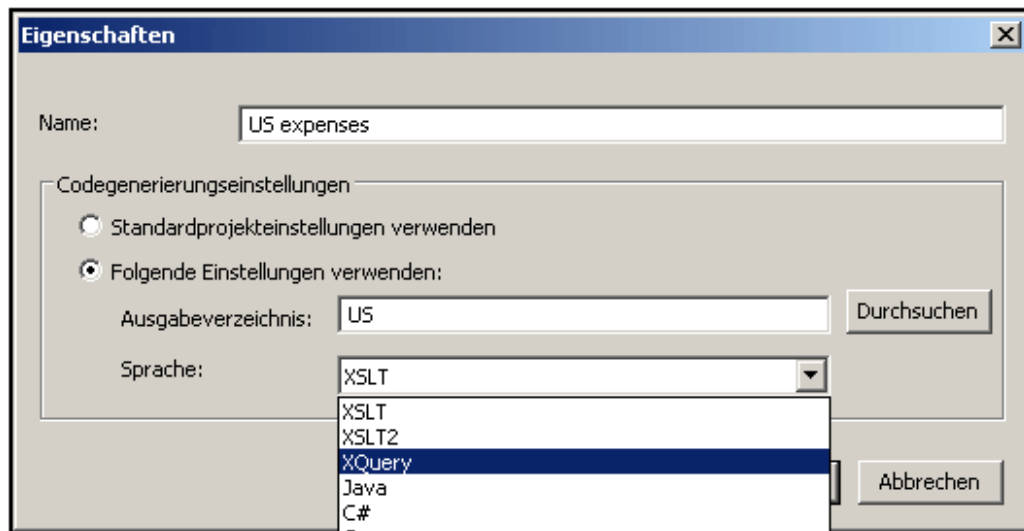
Mit der Option **Dateien zu Projekt hinzufügen...** wird das Windows-Standarddialogfeld **Öffnen** aufgerufen, in dem Sie MapForce Mappings (*.mfg-Dateien) auswählen können, die zum Projekt hinzugefügt werden sollen.



Mit dem Befehl **Aktive Datei zum Projekt hinzufügen** wird das Mapping, das im Mapping-Fenster gerade aktiv ist, zum Mapping hinzugefügt. Wenn die Datei bereits zum Projekt hinzugefügt wurde, wird eine Warnmeldung angezeigt.



Mit der Option **Ordner erstellen** wird das Dialogfeld **Eigenschaften** geöffnet, wo Sie einen Ordernamen definieren und auswählen können, ob Sie die Standardprojekteinstellungen verwenden oder eigene Einstellungen für das Ausgabeverzeichnis und die Sprache vornehmen möchten. Wenn Sie auf **OK** klicken, wird im aktuell ausgewählten Ordner ein Unterordner angelegt.



Bei Auswahl der Option **Projekt | Dateien zu Projekt hinzufügen** können Sie Dateien hinzufügen, die derzeit in MapForce nicht geöffnet sind.

So fügen Sie Mappings zu einem Projekt hinzu:

1. Wählen Sie **Projekt | Aktive Datei zu Projekt hinzufügen**. Daraufhin wird die aktuell aktive Datei zum Projekt hinzugefügt. Der Name des Mappings wird nun auf dem Projektregister unterhalb des Projektnamens angezeigt.

So löschen Sie ein Mapping aus einem Projekt:

Wählen Sie eine der folgenden Methoden:

- Klicken Sie mit der rechten Maustaste auf das Mapping-Symbol unterhalb des Projektordners und wählen Sie im Kontextmenü den Befehl **"Löschen"**.
- Wählen Sie ein Mapping im Projektordner aus und drücken Sie die **Entf**-Taste.

Projekteinstellungen

Das Dialogfeld **Projekteinstellungen** kann durch Rechtsklick auf den Projektnamen auf dem Register "Projekt" und Auswahl des Kontextmenübefehls **Eigenschaften** oder über den Menübefehl **Projekt | Eigenschaften** aufgerufen werden.

Projekteigenschaften

Projektname: Tutorial-project

Projekt-Verzeichnis: Documents\Altova\MapForce2011\MapForceExamples\

Ausgabeeinstellungen

Ausgabename: Tutorial-project

Ausgabeverzeichnis: C:\Documents and Settings\anf\My Documents\Altova\MapForce\Tutorial-project\ Durchsuchen

Sprache: XSLT

Java-Einstellungen

Base Package Name: com.mapforce

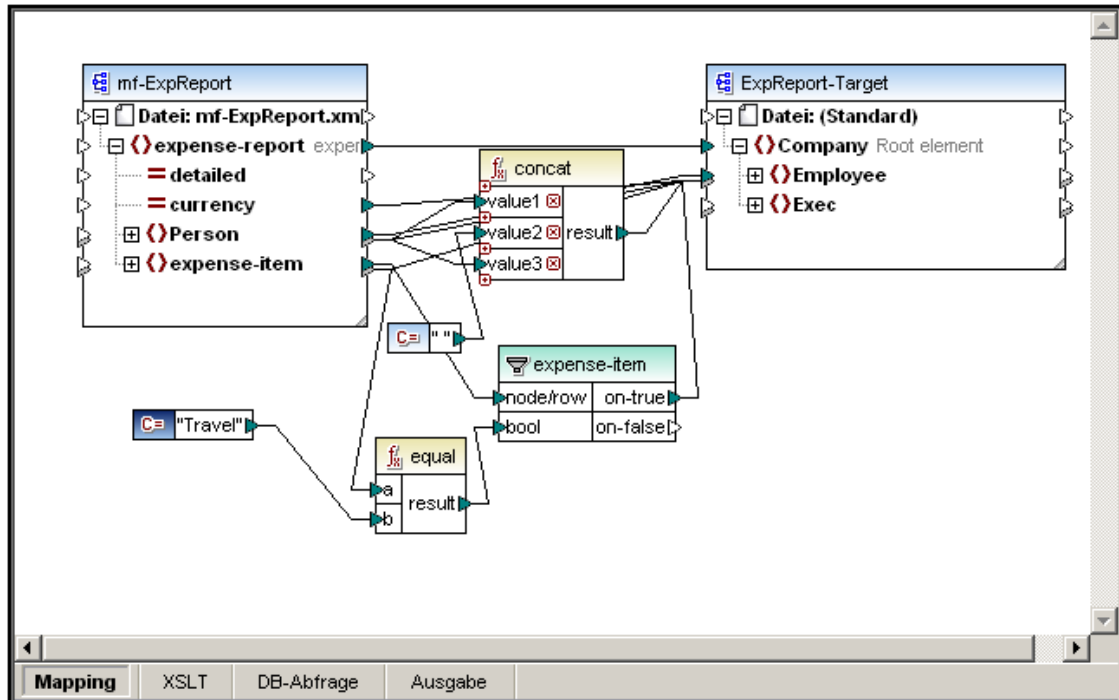
OK Abbrechen

Das Projekt-Verzeichnis ist der Ordner, der die *.mfp-Projektdatei enthält.

Sie können den Ausgabenamen, das Ausgabeverzeichnis und die Standardsprache im Gruppenfeld "Ausgabeeinstellungen" und den Namen des Basispakets (Base package name) im Gruppenfeld Java-Einstellungen ändern. Bitte beachten Sie, dass der Projektname und das Projektverzeichnis nach Erstellung des Projekts nicht mehr geändert werden können.

5.3 Mapping-Fenster

Das Mapping-Fenster ist der Arbeitsbereich, in dem Sie Ihre Mappings in MapForce erstellen.



Im Mapping-Fenster werden die grafischen Elemente angezeigt, mit Hilfe derer das Mapping (die Transformation) zwischen den beiden Komponenten durchgeführt wird. Die Input- und Output-Symbole der einzelnen Schemadaten Elemente werden durch Konnektoren miteinander verbunden. Bei Schema-Datenelementen kann es sich entweder um Elemente oder Attribute handeln.

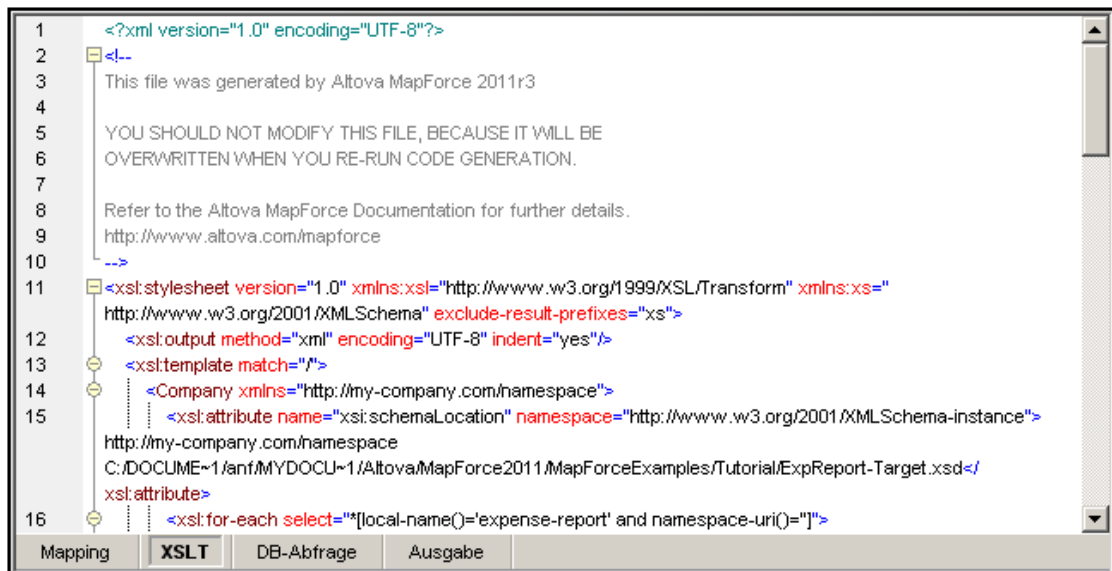
Komponenten mit Hilfe von Hilfslinien ausrichten

Wenn Sie Komponenten im Mapping-Fenster verschieben, werden Hilfslinien zur automatischen Ausrichtung der Komponenten angezeigt, anhand derer Sie die Komponente an jeder beliebigen anderen Komponente im Mapping-Fenster ausrichten können. Diese Option kann über die Menüoption **Extras | Optionen | Allgemein** aktiviert/deaktiviert werden.

In der Abbildung unten wird die untere "value-map"-Komponente verschoben. Die Hilfslinien zeigen an, dass sie an der "contains"-Funktion und an der "ExpReport-item"-Komponente ausgerichtet werden kann.

5.4 Die Fenster XSLT/XSLT2/XQuery

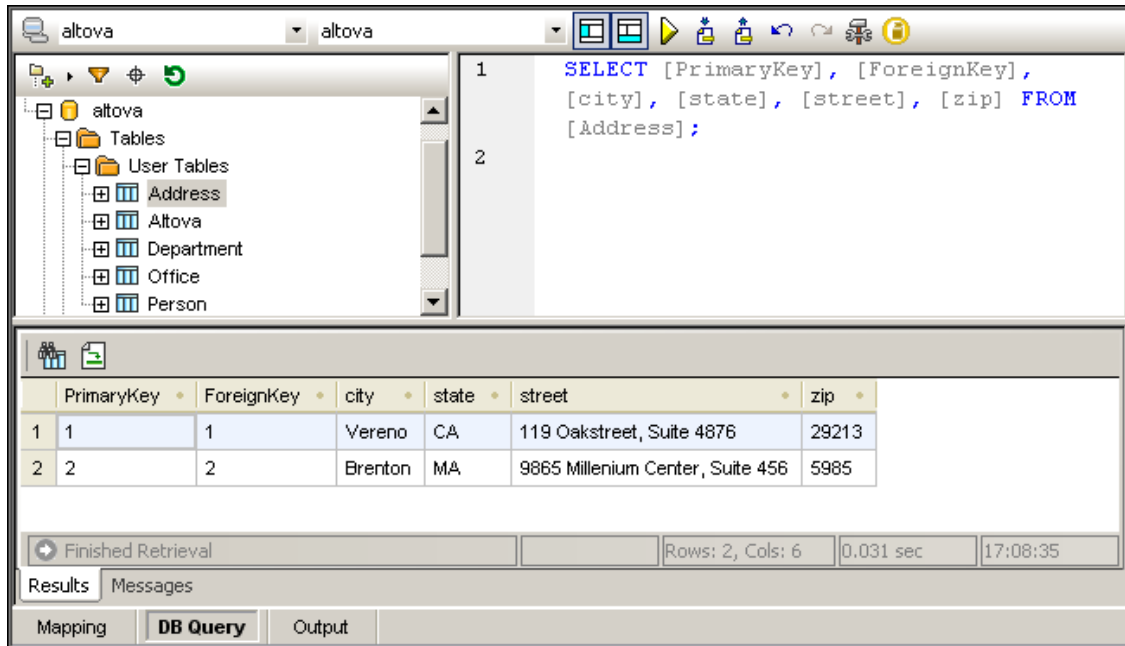
In den Fenstern **XSLT**, **XSLT2** und **XQuery** wird eine Vorschau der jeweils ausgewählten Transformation angezeigt. Die Vorschau hängt von der [jeweils ausgewählten Sprache ab](#).



Anmerkung: Wenn Sie die Ausgabesprache ändern möchten, müssen Sie dazu zuerst in das Mapping-Fenster zurückwechseln. Wenn das Register für eine bestimmte Sprache aktiv ist, können Sie die Ausgabesprache im Menü **Ausgabe** bzw. der Symbolleiste **Auswahl der Sprache** nicht ändern.

5.5 Fenster "DB-Abfrage"

Im Fenster **DB-Abfrage** können Sie jede gängige [Datenbank](#) direkt abfragen. Die hier definierten Abfragen/Aktionen sind unabhängig von allen anderen MapForce-Registern und werden nicht als Teil der *.MFD-Datei gespeichert.



Jedes Datenbankabfragefenster ist mit dem gerade aktiven Mapping verknüpft, sodass **mehrere Datenbankabfragen** pro Sitzung/Mapping durchgeführt werden können. Beachten Sie, dass für jedes Datenbankabfragefenster auch mehrere Verbindungen zu unterschiedlichen Datenbanken aktiv sein können.

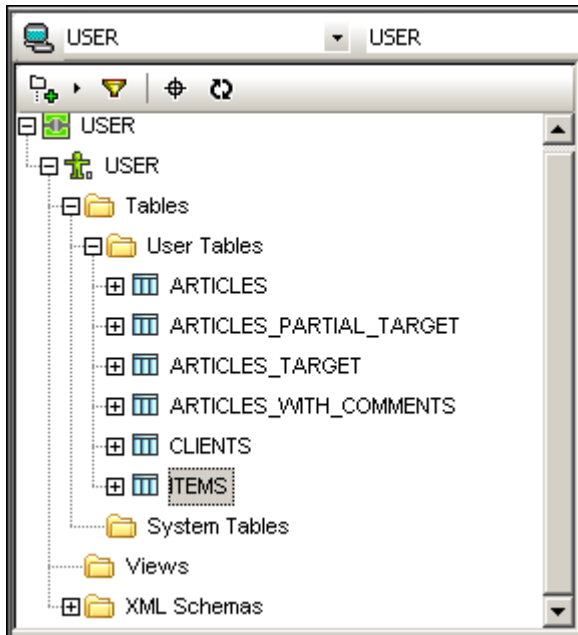
Das Register "Datenbankabfrage" besteht aus den folgenden Fenstern:

- dem [Browser](#)-Fenster auf der linken Seite, in dem die Verbindungsinformationen und Datenbanktabellen angezeigt werden
- das [SQL Editor](#)-Fenster rechts vom Browser-Fenster, in dem Sie Ihre SQL-Abfragen erstellen können
- das [Ergebnisfenster](#), in dem Abfrageergebnisse in tabellarischer Form angezeigt werden
- das [Meldungsfenster](#), in dem Warn- und Fehlermeldungen angezeigt werden

In der obersten Zeile des Datenbankabfragefensters werden die Steuerelemente für die Verbindungen angezeigt. Hier können Sie die Arbeitsdatenbanken sowie die Verbindung und die Datenbankschemas definieren.

5.5.1 Browser-Fenster

Zu jeder der (vielen) verbundenen Datenquellen sehen Sie im **Browser**-Fenster eine Gesamtübersicht der Objekte in den einzelnen Datenbanken, einschließlich Datenbank Constraint-Informationen, z.B. Spalten als Primär- oder Sekundärschlüssel. In Datenbanken vom Typ IBM DB2 Version 9 werden im Browser zusätzlich registrierte XML-Schemas in einem separaten Ordner angezeigt.



Das Browser-Fenster kann angepasst werden um:

- bei Anzeige von Datenbankobjekten bestimmte Ordner-**Layouts** anzuzeigen
- bestimmte Objekte in der Datenbank mit Hilfe der Objektsuche zu **suchen**
- die Anzahl der angezeigten Objekte zu **filtern**
- das Root-Objekt der aktiven Datenquelle zu **aktualisieren**

Ordner-Layouts

Das Browser-Fenster enthält eine Reihe von vordefinierten Layouts zur Anzeige verschiedener Datenbankobjekte:

- Im Layout **Ordner** werden Datenbankobjekte auf Basis des Objekttyps in einer hierarchischen Struktur in Ordner unterteilt. Dies ist die Standardeinstellung.
- Das Layout **Keine Schemas** ähnelt dem Ordner-Layout, mit der Ausnahme, dass es keine Datenbankschema-Ordner gibt; Tabellen werden daher nicht nach Datenbankschemas gegliedert.
- Im Layout **Keine Ordner** werden Datenbankobjekte in einer Hierarchie ohne Verwendung von Ordnern angezeigt.
- Im Layout **Flach** werden Datenbankobjekte nach ihrem Typ auf der ersten hierarchischen Ebene geordnet. So werden Spalten z.B. anstatt in der entsprechenden Tabelle in einem separaten Ordner für Spalten angezeigt.
- Im Layout **Tabellenabhängigkeiten** werden Tabellen nach ihren Beziehungen zu anderen Tabellen gegliedert. Es gibt Kategorien für Tabellen mit Sekundärschlüsseln, durch Sekundärschlüssel referenzierte Tabellen und Tabellen, die in keiner Beziehung

zu anderen Tabellen stehen.

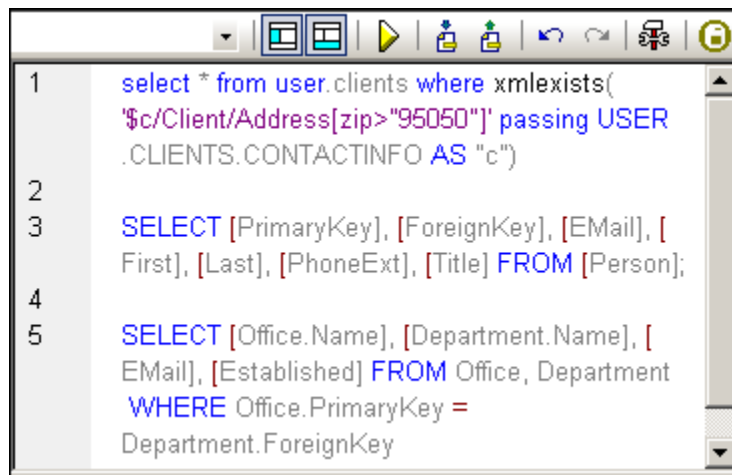
Im Standardlayout "Ordner" werden Datenbankobjekte hierarchisch angezeigt. Je nachdem, welches Objekt ausgewählt ist, stehen bei Rechtsklick auf dieses Objekt unterschiedliche [Kontextmenüoptionen](#) zur Verfügung.

5.5.2 SQL Editor

Der **SQL Editor** dient zum Schreiben und Ausführen von SQL-Anweisungen.

SQL Editor-Funktionen:

- [automatischen Generierung](#) von SQL-Anweisungen mittels **Drag&Drop** aus dem Browser-Fenster
- [automatische Kompletierung](#) von SQL-Anweisungen beim Erstellen von SELECT-Anweisungen
- Definition von [Regionen](#)
- Einfügen von Zeilen- oder Block-[Kommentaren](#)



Die SQL-Symbolleiste enthält die folgenden Schaltflächen:



Browser ein/aus: Blendet das Browser-Fenster ein und aus.



Ergebnisse ein/aus: Blendet das Ergebnisfenster ein und aus.



Ausführen (F5): Wenn Sie auf diese Schaltfläche klicken, werden die gerade im aktiven Fenster des SQL Editors ausgewählten SQL-Anweisungen ausgeführt. Wenn es mehrere Anweisungen gibt und keine ausgewählt ist, werden alle ausgeführt.



Rückgängig: Dient zum Rückgängigmachen einer unbegrenzten Anzahl von Bearbeitungen im SQL-Fenster.



Wiederherstellen: Dient zum Wiederherstellen von zuvor rückgängig gemachten Befehlen. Mit Hilfe dieser beiden Befehle können Sie sich im Verlauf vor und zurückbewegen.



SQL-Datei importieren: Öffnet eine SQL-Datei im SQL-Editor, die dann ausgeführt werden kann.



SQL-Datei exportieren: Speichert SQL-Abfragen zur späteren Wiederverwendung



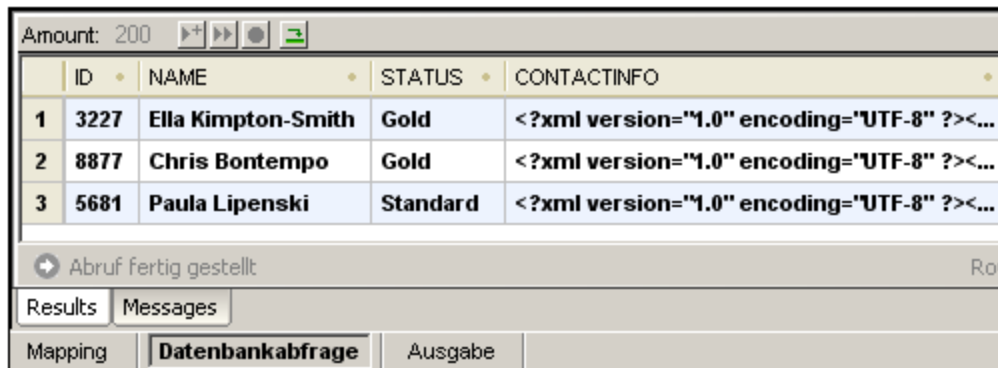
SQL Script in DatabaseSpy öffnen: Startet DatabaseSpy und öffnet das Script im SQL Editor-Fenster.



Optionen: Öffnet das Dialogfeld "Optionen", in dem Sie sowohl allgemeine als auch SQL-Editor-Einstellungen definieren können.

5.5.3 Register "Ergebnisse"

Auf dem **Ergebnisregister** des SQL Editors sehen Sie den als Ergebnis der Datenbankabfrage abgerufenen Datensatz.



	ID	NAME	STATUS	CONTACTINFO
1	3227	Ella Kimpton-Smith	Gold	<?xml version="1.0" encoding="UTF-8" ?><...
2	8877	Chris Bontempo	Gold	<?xml version="1.0" encoding="UTF-8" ?><...
3	5681	Paula Lipenski	Standard	<?xml version="1.0" encoding="UTF-8" ?><...

Abruf fertig gestellt

Results Messages

Mapping Datenbankabfrage Ausgabe

Symbolleistenoptionen - Abrufmodus

Das Ergebnisfenster enthält eine Symbolleiste, mittels der Sie zwischen den Ergebnissen und SQL-Anweisungen navigieren können und Teile von Datenbankdaten einfacher abrufen können.



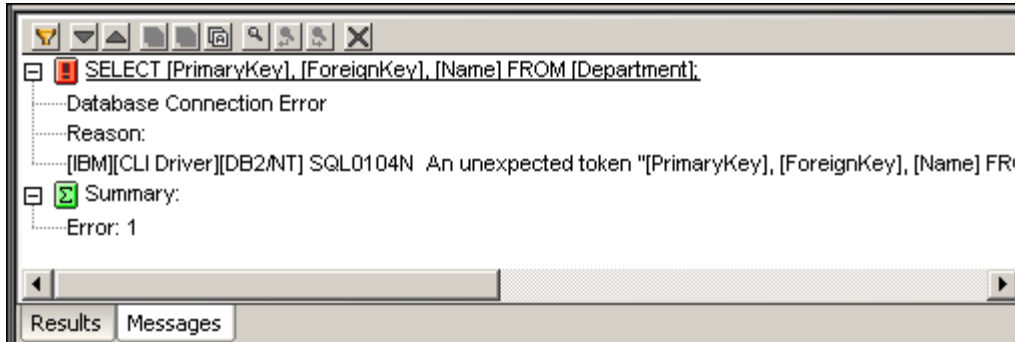
Suchen: Durchsucht den Input-String im Ergebnisfenster. Mit Hilfe der Funktionstaste F3 können Sie die Suche fortsetzen.



Gehe zu Anweisung: Geht zum SQL Editor-Fenster und markiert die Gruppe von SQL-Anweisungen, die das aktuelle Ergebnis erzeugt hat..

5.5.4 Register "Meldungen"

Auf dem **Meldungsregister** des SQL Editors werden bestimmte Informationen zur zuvor ausgeführten SQL-Anweisung sowie Fehler- und Warnmeldungen angezeigt.



Sie können zum Anpassen der Ansicht des Meldungsregisters verschiedene Filter verwenden oder das Fenster Zeile für Zeile mit Hilfe der Schaltflächen **Nächste** und **Vorherige** durchsuchen. Auf dem Meldungsregister steht außerdem ein Dialogfeld **Suchen** zur Verfügung sowie eine Reihe von Optionen zum Kopieren von Text in die Zwischenablage.

Symbolleistenoptionen

Das Meldungsregister enthält eine Symbolleiste, über die Sie innerhalb der Meldungen navigieren können und die Filter zum Ein- und Ausblenden bestimmter Teile der Meldung enthält.



Filter: Wenn Sie auf diese Schaltfläche klicken, wird ein Popup-Menü aufgerufen, in dem Sie einzelne Meldungsteile auswählen können (**Übersicht, Erfolg, Warnung, Fehler**). Des Weiteren können Sie alle oder keiner der Optionen mit einem einzigen Mausklick aktivieren bzw. deaktivieren. Wählen Sie einfach aus dem Popup-Menü eine der Optionen **Alle aktivieren** oder **Alle deaktivieren**.



Nächste: Springt zur nächsten Meldung und markiert sie.



Vorherige: Springt zur vorherigen Meldung und markiert sie.



Ausgewählte Meldung in die Zwischenablage kopieren



Ausgewählte Meldung einschließlich ihrer Subeinträge in die Zwischenablage kopieren



Alle Meldungen in die Zwischenablage kopieren



Suchen: Ruft das Dialogfeld **Suchen** auf.



Vorheriges suchen: Springt zur vorhergehenden Instanz des im Dialogfelds **Suchen** definierten String.



Weitersuchen: Springt zur nächsten Instanz des im Dialogfelds **Suchen** definierten String.



Löschen: Entfernt alle Meldungen vom Meldungsregister des SQL Editor-Fensters.

Bitte beachten Sie:

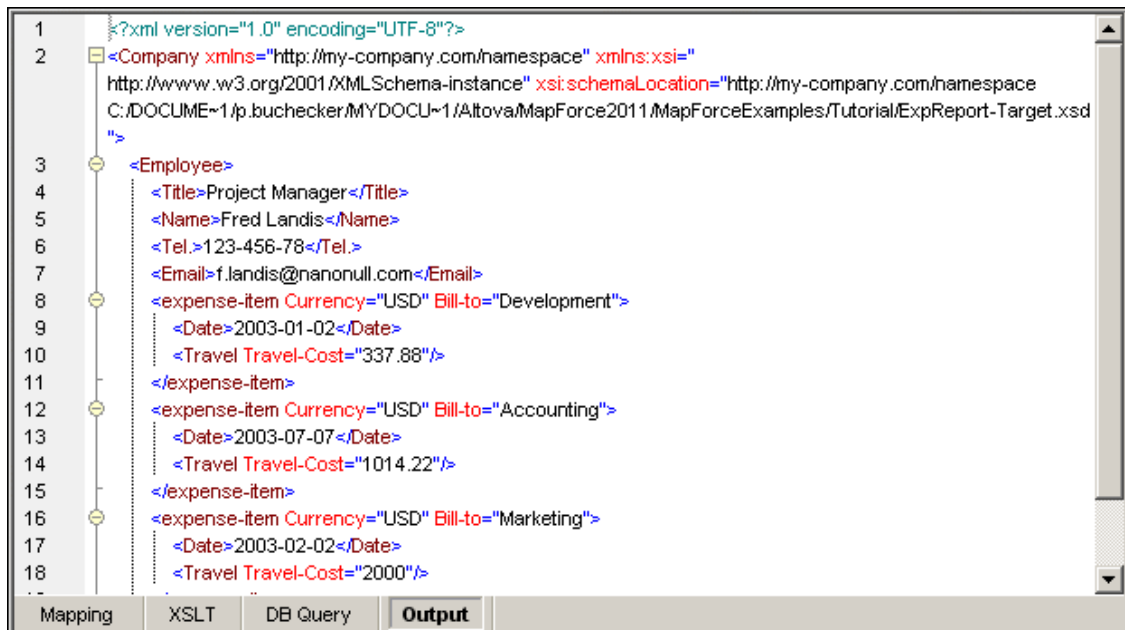
Dieselben Optionen stehen im Kontextmenü des Ergebnisfensters zur Verfügung.

5.6 Fenster "Ausgabe" / BUILTIN-Ausführungsprozessor

Das Mapping-Ergebnis wird sofort, je nachdem welche Sprache ausgewählt ist, unter Verwendung des Altova XSLT oder XQuery-Prozessors im Ausgabe-Fenster dargestellt.

Wenn eine der kompilierten Programmiersprachen (Java, C++ oder C#) oder [BUILTIN](#) ausgewählt ist, generiert der Built-in-Ausführungsprozessor die Ausgabe anhand der Quellkomponenten: XML/Schema-Dateien, Textdateien, Datenbanken usw. Das Ergebnis, das im Fenster "Ausgabe" angezeigt wird, ist dasselbe, das Sie bei Generierung, Kompilierung und Ausführung des Java-, C++- oder C#-Codes erhalten.

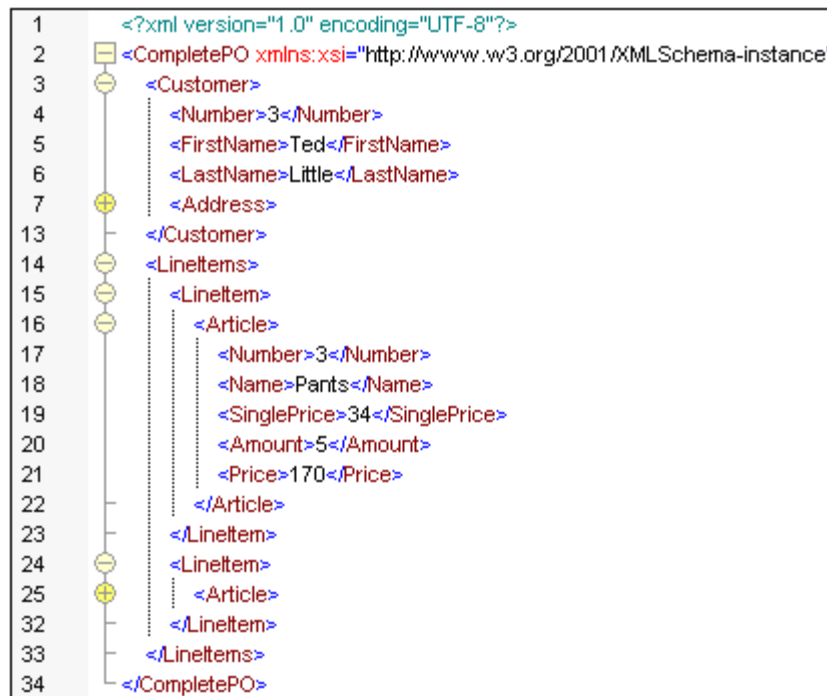
Bitte beachten Sie: Beim vom Built-in-Ausführungsprozessor generierten Ergebnis handelt es sich um eine on-the-fly-Transformation von Datenbank-, Text-, oder EDI-Daten, ohne dass Sie Programmcode generieren oder kompilieren müssen! Wir empfehlen Ihnen, diese Option zu verwenden, bis Sie mit dem Ergebnis zufrieden sind und anschließend erst Programmcode zu generieren.



Je nach **Zielkomponente** Ihres Mappings, werden im Ausgabe-Fenster unterschiedliche Dinge angezeigt:

- **XML Schema/Dokument als Ziel:**

Im unten gezeigten Screenshot sehen Sie das Ergebnis der Mappings **DB_CompletePO.mfd** aus dem Ordner [...MapForceExamples](#). In diesem Mapping werden sowohl ein XML-Schema/Dokument als auch eine Datenbank als Quellkomponenten verwendet.



Die erzeugte XML-Datei lässt sich durch Klicken auf das Symbol **Generierte Ausgabe speichern**  speichern, und durch einen Klick auf das Symbol **Ausgabe validieren**  in der Symbolleiste anhand des referenzierten Schemas validieren.

- **Datenbank als Ziel:**

Führt das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.

```

/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
Provider=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Documents and Settings\...My Documents\
Altova\MapForce2012\MapForceExamples\altova.mdb;
*/

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name],[PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')

```

Durch Klicken auf das Symbol **SQL-Skript ausführen**  führen Sie die SQL select-Anweisung aus und erhalten einen Bericht zu den Datenbank-Aktionen, wie unten beschrieben.

- die eigentliche SQL-Anweisung, die in der Zieldatenbank ausgeführt wurde.

```
SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
```

```
-->>> OK. One or more rows.
```
- Mehrfache Tabellenaktionen, falls welche ausgeführt wurden,

```
INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', 1)
-->>> OK. 1 row(s).
```

- die Ergebnisse jeder SQL-Anweisung:
-->>> OK. n row(s). , wenn die Anweisung erfolgreich ausgeführt wurde oder Execution failed und eine detaillierte Fehlermeldung.

```
SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> OK. One or more rows.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 row(s).
```

- Textdatei als Ziel:**

Der Built-in-Ausführungsprozessor bietet Unterstützung für die Anzeige der Ergebnisse der folgenden Textdateien als Ziel:

- CSV-Dateien (kommagetrennte Werte) - auch für andere Trennzeichen als Kommas
- FLF-Dateien (fixed-length fields)

1	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Albert,Aldrich,582,A.Ald
2	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Bert,Bander,471,b.bander
3	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clovis,963,c.clovi
4	"Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Dave,Durne
5	"Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Eve,Ellas,
6	"Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Fred,Fortunat,95
7	"Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Gerry,Gundall,65

- UN/EDIFACT- und ANSI X12-Dateien
- eine FlexText-Vorlage, in der die Dateistruktur und der als Ziel verwendete Inhalt definiert sind

- MS OOXML Excel 2007 (und höher)-Dateien als Ziel**

Microsoft Excel 2007 (und höher) muss nur installiert sein, wenn Sie eine **Vorschau** von Excel 2007 (und höher)-Blättern auf dem Register **Ausgabe** anzeigen wollen. Wenn Sie eine frühere Version von Microsoft Excel verwenden oder Excel nicht installiert haben, können Sie trotzdem von/auf Excel 2007 (und höher)-Dateien mappen. Sie können das Ergebnis nicht auf dem Register "Ausgabe" anzeigen, können es aber durch Klicken auf die Schaltfläche "Ausgabe speichern" speichern.

	A	B	C	D
1	Alex	Martin		
2	George	Hammer		
3	Jessica	Bander		
4	Lui	King		
5				
6				

Tastaturkürzel für das Fenster Ausgabe (Tastatur und Ziffernblock):

Strg und "+" in den Text hineinzoomen
 Strg und "-" aus dem Text herauszoomen
 Strg und "0" setzt den Zoomfaktor auf die Standardeinstellung zurück

Strg und Mausrad nach vorne / nach hinten erzielt den gleichen Ein- und Auszoom-Effekt.

5.7 Fenster im Zusammenhang mit StyleVision

Auf den Registern **HTML**, **RTF**, **PDF** und **Word 2007+** werden die Daten der Zielkomponente als HTML-, RTF-, PDF- oder Word 2007+-Dokument dargestellt, wenn ein [StyleVision Power Stylesheet \(SPS\)](#) mit der Zielkomponente verknüpft ist.

Nanonull Inc. Nanonull

Street:	900 Cummings Center
City:	Beverly
State & Zip:	MA 01915
Phone:	+1 (321) 555 5155
Fax:	+1 (321) 555 5155 - 9
E-mail:	office@nanonull.com

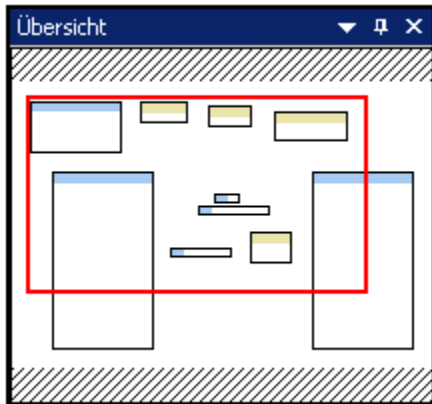
Employees

Last Name	First Name	Details
Callaby	Vernon	

Mapping XSLT2 DB-Abfrage Ausgabe **HTML** RTF PDF Word 2007+

5.8 Fenster "Übersicht"

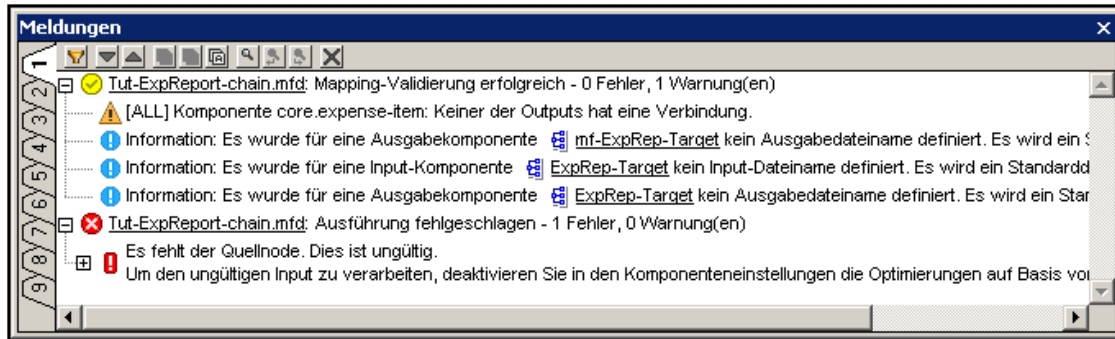
Das Fenster "Übersicht" dient als Navigationsfenster für große Mappings. Der im Mapping-Fenster sichtbare Bereich wird im Fenster "Übersicht" durch ein rotes Rechteck markiert. Sie können das Rechteck im Fenster "Übersicht" mit der Maus ziehen, um den entsprechenden Bereich im Mapping-Fenster zu sehen.



Durch Klicken in das Fenster "Übersicht" wird der Mittelpunkt der Anzeige im Mapping-Fenster definiert.

5.9 Fenster "Meldungen"

Im Fenster "Meldungen" werden Meldungen, Fehler und Warnmeldungen angezeigt, wenn Sie auf die Schaltfläche "Ausgabe" klicken oder eine Mapping-Validierung ausführen.

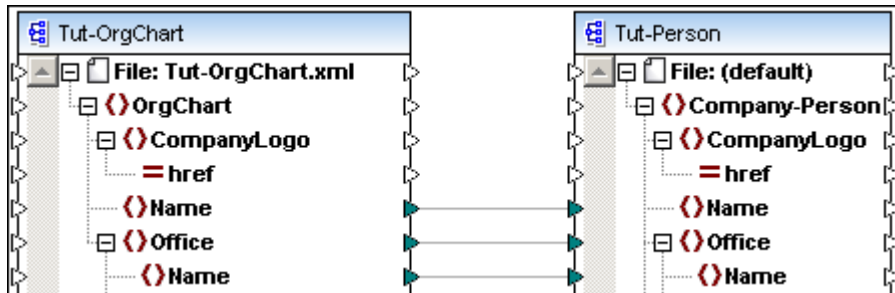


Kapitel 6

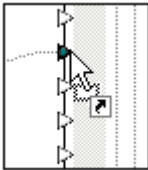
Mapping zwischen Komponenten

6 Mapping zwischen Komponenten

Ein **Konnektor** visualisiert das **Mapping** zwischen zwei Daten-Sets und dient dazu die **Quelldaten** (den Wert) in das Zielschema/-dokument bzw. die Zieldatenbank zu transformieren, d.h. zu übernehmen.



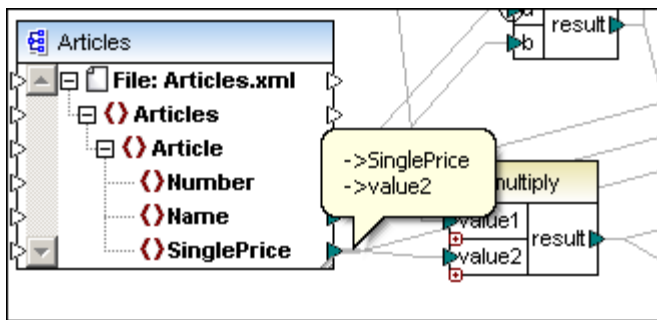
Komponenten und **Funktionen** weisen kleine "Verbindungs-dreiecke", die so genannten **Input-** oder **Output-Symbole**. Diese **Symbole** befinden sich links bzw. rechts von allen **Datenelementen**, die gemappt werden können. Wenn Sie auf ein solches Symbol klicken und es mit der Maus ziehen, wird ein **Mapping-Konnektor** erzeugt, den Sie nun auf ein anderes Symbol oder einen Datenelementnamen ziehen können. Neben dem Text-Cursor erscheint ein Verknüpfungssymbol, wenn dieses Mapping zulässig ist.



Wenn Sie auf einen **Datenelementnamen** (Element/Attribut) klicken, wird automatisch das dazugehörige **Quellsymbol** für die Verknüpfung ausgewählt. Wenn Sie den Konnektor auf den Namen des Zieldatenelements ziehen, wird das Zielsymbol automatisch ausgewählt.

Ein **Input-Symbol** kann nur einen Konnektor haben. Wenn Sie versuchen, einen zweiten Konnektor damit zu verbinden, werden Sie gefragt, ob Sie das Input-Symbol ersetzen oder **duplizieren** möchten. Ein **Output-Symbol** kann mehrere Konnektoren haben, die mit unterschiedlichen Input-Symbolen verbunden sind.

Wenn Sie den Mauszeiger über den geraden Abschnitt eines Konnektors platzieren (in der Nähe des Input/Output-Symbols), wird dieser markiert und es erscheint ein Popup-Fenster. Darin sehen Sie den/die Namen des/der Datenelement(e) am anderen Ende des Konnektors. Wenn mehrere Konnektoren vom selben Output-Symbol aus definiert wurden, werden maximal zehn Datenelementnamen angezeigt. In der Abbildung sehen Sie, dass die beiden Ziel-Datenelemente **SinglePrice** und **value2** der multiply-Funktion sind.



Wenn Sie auf den geraden Abschnitt des Konnektors klicken, wird er markiert, sodass Sie ihn mit der Maus an eine andere Stelle ziehen können.

Um einen **duplizierten Konnektor** von derselben Quelle zu einem anderen Ziel zu erstellen, positionieren Sie den Mauszeiger in der Nähe des ursprünglichen Ziels über den geraden Abschnitt des Konnektors und ziehen Sie den Konnektor auf ein anderes Ziel, während Sie die Strg-Taste gedrückt halten.

So suchen Sie in einer Komponente nach einem bestimmten Node/Datenelement:

1. Klicken Sie auf die gewünschte Komponente und drücken Sie die Tasten Strg+F.
2. Geben Sie den Suchbegriff ein und klicken Sie auf Weitersuchen.

Über die oben in der Abbildung angezeigten erweiterten Optionen können Sie festlegen, welche Datenelemente/Nodes durchsucht werden sollen. Außerdem können Sie die Suchoptionen auf Basis der Verbindungen einschränken.

Automatisches Verbindung von Datenelementen

Wenn Sie auf die Schaltfläche "Sub-Einträge automatisch verbinden"  klicken und einen Konnektor zwischen den beiden Elementen erstellen, werden alle Sub-Einträge desselben Namens, die sich unter dem übergeordneten Datenelement befinden, automatisch verbunden.

Anzahl der Konnektoren

Input- und Output-Symbole werden an den meisten Komponenten angezeigt. Es besteht jedoch zwischen der Anzahl dieser Symbole keine 1:1-Beziehung.

- Jedes **Schemaelement** (Element/Attribut) hat ein Input- und ein Output-Symbol.

- **Datenbankelemente** haben Input- und Output-Symbole.
- Schema-, Datenbank- und andere Komponenten in benutzerdefinierten Funktionen haben nur Output-Symbole.
- Duplizierte Datenelemente haben nur Input-Symbole. Auf diese Art können mehrere Eingaben auf sie gemappt werden. Nähere Informationen dazu finden Sie unter "[Duplizieren der Eingabeelemente](#)".
- **Funktionen** können beliebig viele Input- und Output-Symbole haben, und zwar für jeden **Parameter eines**. So hat z.B. die Funktion "Add" zwei (oder mehr) Input-Symbole und ein Output-Symbol.
- **Spezielle** Komponenten können unterschiedlich viele Symbole haben. So hat z.B. die Komponente "Konstante" nur ein Output-Symbol.

6.1 Datenmappingmethoden (Standard / Mixed Content / Sub-Einträge kopieren)

MapForce unterstützt verschiedene Methoden, Daten zu mappen: [Zielorientiert \(Standard\)](#), [quellorientiert \(Mixed Content\)](#) und [Alles kopieren \(Sub-Einträge kopieren\)](#).

Konnektoren und ihre Eigenschaften:

Die folgenden Aktionen können - mit den beschriebenen Ergebnissen - an einem Konnektor ausgeführt werden:

- Durch Klicken auf einen Konnektor wird dieser rot markiert.
- Durch Drücken der **Entf**-Taste, wird der Konnektor sofort gelöscht, wenn er markiert ist.
- Durch Rechtsklicken auf einen Konnektor wird das Konnektor-Kontextmenü geöffnet.
- Durch Doppelklick auf einen Konnektor wird das Dialogfeld "[Verbindungseinstellungen](#)" geändert.

Anzeigen von Konnektoren

In MapForce können Sie die Konnektoren im Mapping-Fenster selektiv anzeigen.



Ausgewählte Komponenten-Konnektoren anzeigen

Wechselt zwischen der Anzeige:

- aller Mapping-Konnektoren in Schwarz und
- der Komponenten, die mit der aktuell ausgewählten Komponente verknüpft sind (in Schwarz). Andere Konnektoren erscheinen abgeblendet.



Konnektoren zwischen Quelle und Ziel anzeigen

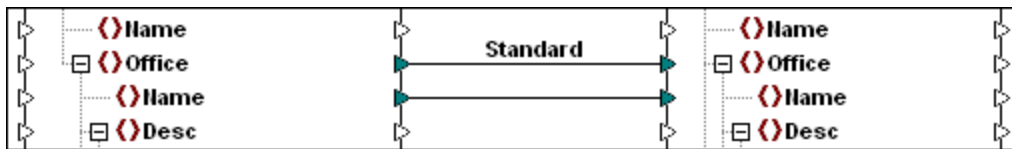
Wechselt zwischen der Anzeige von:

- Konnektoren, die mit der aktuell ausgewählten Komponente **direkt** verbunden sind und
- Konnektoren, die mit der aktuell ausgewählten Komponente verknüpft sind, von einer Quellkomponente ausgehen und diese mit den Zielkomponenten verbinden.

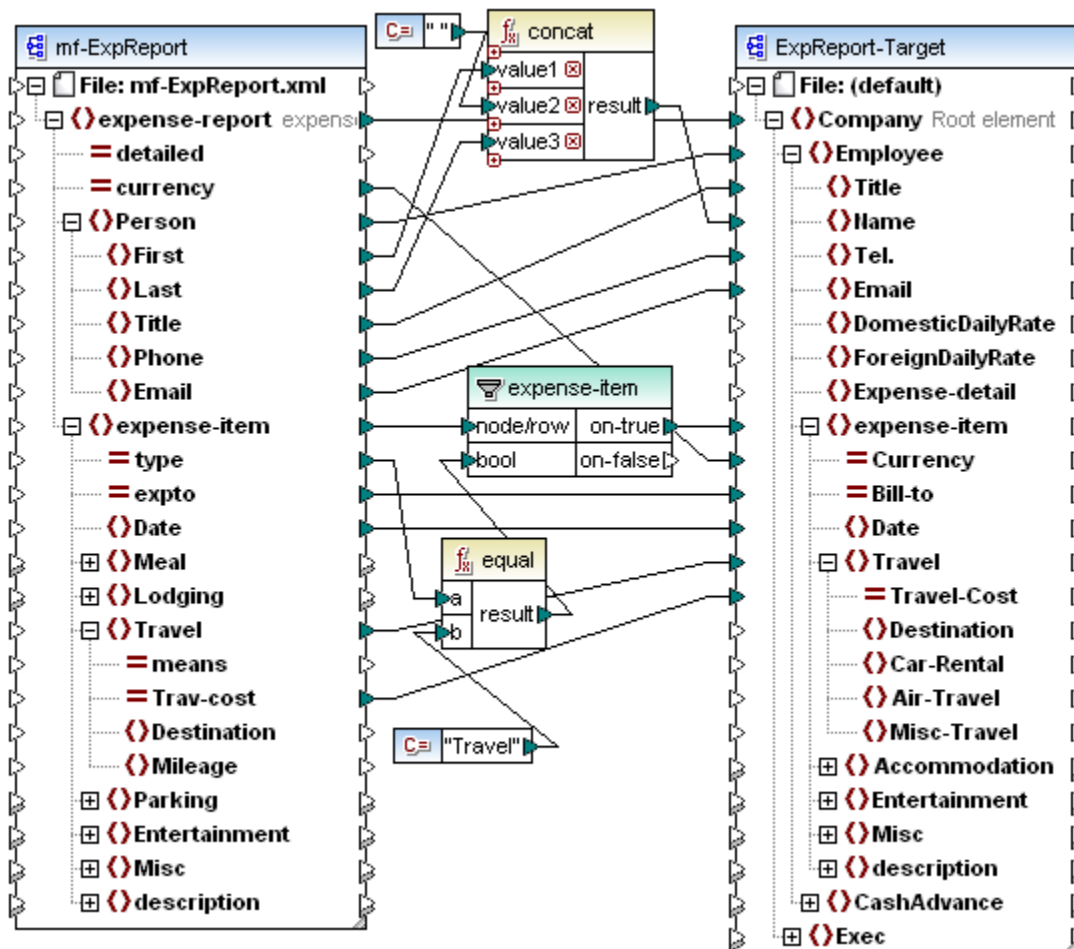
6.1.1 Zielorientiert / Standard-Mapping

Ein Standardmapping ist die normalerweise in MapForce verwendete Mappingmethode, d.h. die Ausgabe ist von der Reihenfolge der Zielnodes abhängig. Nähere Informationen finden Sie unter [Quellorientiertes / Mixed Content Mapping im Gegensatz zum Standardmapping](#).

- Der Inhalt von Mixed Content Text Nodes wird **nicht** unterstützt/gemappt.
- Die Reihenfolge der Child Nodes ist von der **Ziel-Schema**-Datei abhängig.



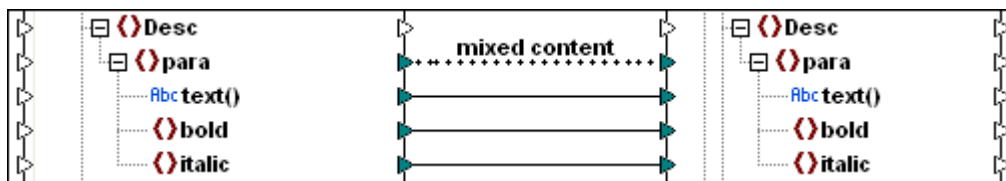
Standard-Mappings werden durch eine durchgezogene Linie gekennzeichnet.



6.1.2 Quellorientiertes / Mixed Content Mapping

In einem quellorientierten (Mixed Content) Mapping können Text-Nodes und Child Nodes automatisch in derselben Reihenfolge, in der sie in der XML-**Quelldatei** vorkommen, gemappt werden.

- Der Inhalt von Mixed Content Text-Nodes wird unterstützt/gemappt.
- Die Reihenfolge von Child Nodes ist abhängig von der **Quell**-XML-Instanzdatei.



Mixed Content-Mappings werden durch eine gepunktete Linie gekennzeichnet.

Quellorientiertes Mapping kann natürlich auf Wunsch auch auf XML-Schema-Datenelemente vom Typ **complexType** angewendet werden. Die Child Nodes werden dann entsprechend ihrer Reihenfolge in der XML-Quelldatei gemappt.

Quellorientiertes / Mixed Content Mapping unterstützt:

- als **Quellkomponenten**:
 - XML-Schema complexTypes (darunter auch Mixed Content, d.h. mixed=true)
 - XML-Schema complexTypes (darunter auch Mixed Content in eingebetteten Schemas eines Datenbankfelds)
- als **Zielkomponenten**:
 - XML-Schema complexTypes (einschließlich Mixed Content),
 - XML-Schema complexTypes (darunter auch Mixed Content in eingebetteten Schemas eines Datenbankfelds)

Anmerkung: CDATA-Abschnitte werden als Text behandelt.

Mappen von Mixed Content

Sie finden die im folgenden Beispiel (**Tut-OrgChart.mfd**) verwendeten Dateien im Ordner [...MapForceExamples\Tutorial\](#).

XML-Quellinstanz:

Im unten gezeigten Screenshot sehen Sie einen Ausschnitt aus der in diesem Abschnitt verwendeten XML-Datei Tut-OrgChart. Hier geht es um das Mixed Content-Element "para" mit seinen Child Nodes "bold" und "italic".

Beachten Sie, dass das Element "para" auch eine Processing Instruction (sort alpha-ascending, also sortiere in aufsteigend in alphanumerischer Reihenfolge) sowie einen Comment-Text (Company details...) enthält, die ebenfalls gemappt werden können.

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp2 U (http://www.altova.com) by Mr. Nobody (Altova GmbH) -->
<OrgChart xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Tut-OrgChart.xsd">
  <CompanyLogo href="nanonull.gif"/>
  <Name>Organization Chart</Name>
  <Office>
    <Name>Nanonull, Inc.</Name>
    <Desc>
      <para>The company was established in<b>Vereno</b>in 1995. Nanonull
develops nanoelectronic technologies for<i>multi-core processors.</i>February 1999
saw the unveiling of the first prototype <b>Nano-grid.</b>The company hopes to expand
its operations <i>offshore</i>to drive down operational costs.
      <?sort alpha-ascending?>
      <!-- Company details: location and general company information.-->
    </para>
    <para>White papers and further information will be made available in the near future.
  </Desc>

```

Beachten Sie bitte die **Reihenfolge** der Text-Nodes und der bold/italic Nodes von Nanonull., Inc in the XML-Instanzdatei. Sie lautet wie folgt:

```

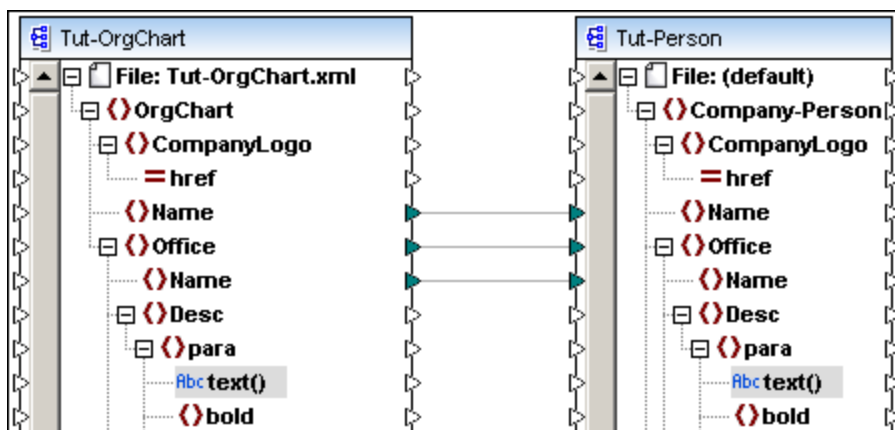
<para> The company...
  <b>Vereno</b>in 1995 ...
  <i>multi-core...</i>February 1999

  <b>Nano-grid.</b>The company ...
  <i>offshore...</i>to drive...
</para>

```

Anfangsmapping

Unterhalb sehen Sie das Mapping in seinem Anfangszustand, wie Sie es sehen, wenn Sie die Datei Tut-Orgchart.mfd öffnen.



Ausgabe des obigen Mappings:

Unterhalb sehen Sie das Ergebnis dieses ersten Mappings: Es wurden Organization Chart sowie die Namen der einzelnen Büros ausgegeben.

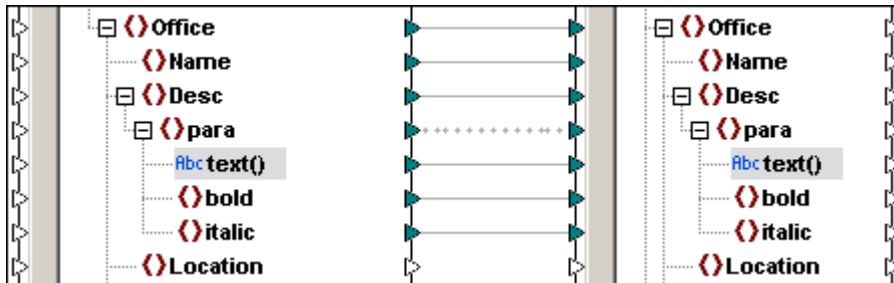
```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company-Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNames
3  <Name>Organization Chart</Name>
4  <Office>
5  <Name>Nanonull, Inc.</Name>
6  </Office>
7  <Office>
8  <Name>Nanonull Europe, AG</Name>
9  </Office>
10 </Company-Person>
11

```

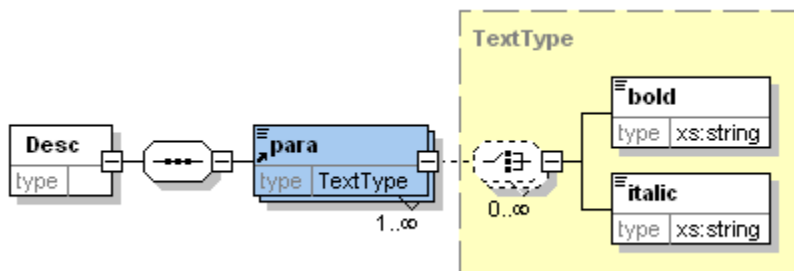
Mappen des para-Elements

In der nachfolgenden Abbildung sehen Sie ein Beispiel für ein Mixed Content Mapping. Das Element "para" hat Mixed Content, daher wird die Verbindungslinie **gepunktet** angezeigt. Der Node **text()** enthält Textdaten und muss gemappt werden, damit der Text in der Zielkomponente aufscheint.



Wenn Sie mit der rechten Maustaste auf einen Konnektor klicken und den Eintrag "Eigenschaften" wählen, können Sie einen Annotationstext zum Konnektor hinzufügen. Nähere Informationen dazu finden Sie im Abschnitt "Referenz" unter "[Verbindung](#)".

In der Abbildung unten sehen Sie das Content Model des Elements "Description" (Desc) der Schema-Datei **Tut-OrgChart.xsd**. Diese Definition ist in dem in diesem Beispiel verwendeten Quell- und Zielschema identisch.



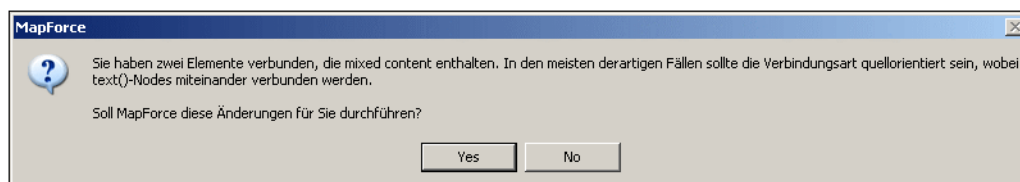
Beachten Sie, dass das Element **para** im Content Model die folgenden Eigenschaften hat:

- para ist ein complexType mit **mixed** = true vom Typ TextType.
- Die Elemente **"bold"** (fett) und **"italic"** (kursiv) sind beide vom Typ **xsd:string**. Sie wurden in diesem Beispiel nicht als rekursiv definiert, d.h. sind weder "bold" noch "italic" vom Typ "TextType".
- Die Elemente **"bold"** und **"italic"** können beliebig oft in beliebiger Reihenfolge innerhalb von **"para"** vorkommen.
- Innerhalb des Elements **"para"** kann eine beliebige Anzahl von Text Nodes vorkommen, die eine beliebige Anzahl der Elemente **"bold"** und **"italic"** in beliebiger Reihenfolge enthalten können.

Erstellen von Mixed Content-Verbindungen zwischen Datenelementen:

1. Wählen Sie die Menüoption **Verbindung | Idente Sub-Einträge automatisch verbinden**, um diese Option zu aktivieren, falls sie noch nicht aktiv ist.
2. Verbinden Sie das Datenelement **para** im Quellschema mit dem Datenelement **para** im Zielschema.

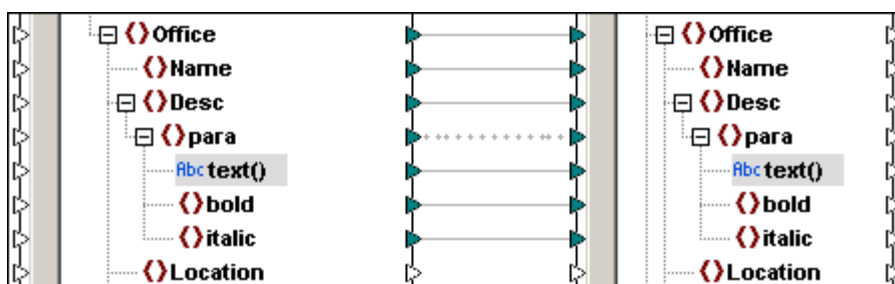
Es erscheint eine Meldung, in der Sie gefragt werden, ob MapForce die Konnektoren als "quellorientiert" definiert werden sollen.



3. Klicken Sie auf Ja, um eine Mixed Content-Verbindung zu erstellen.

Bitte beachten Sie:

"para" ist vom Typ Mixed Content. Aus diesem Grund wird die Meldung zu diesem Zeitpunkt angezeigt. Die Mixed Content-Meldung wird auch angezeigt, wenn Sie die Datenelemente "para" direkt mappen, ohne dass die Option zur automatischen Verbindung der Sub-Einträge aktiv ist.



Alle Child-Datenelemente von "para" wurden verbunden. Der Konnektor, der die Datenelemente "para" verbindet wird als gepunktete Linie angezeigt, da es sich hierbei um Mixed Content handelt.

4. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company-Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNames
3    <Name>Organization Chart</Name>
4    <Office>
5      <Name>Nanonull, Inc.</Name>
6      <Desc>
7        <para>The company was established in<b> Vereno</b>in 1995. Nanonull devel
8        </para>
9        <para>White papers and further information will be made available in the near future.
10       </para>
11      </Desc>
12    </Office>
13    <Office>
14      <Name>Nanonull Europe, AG</Name>
15      <Desc>
16        <para>In May 2000, Nanonull<i>Europe</i> was set up in Vienna. The team co
17        </para>
18      </Desc>
19    </Office>
20  </Company-Person>

```

5. Klicken Sie in der Symbolleiste des Ausgaberegisters auf das Symbol **Zeilenumbruch**



, um den gesamten Text im Fenster "Ausgabe" sehen zu können.

```

3    <Name>Organization Chart</Name>
4    <Office>
5      <Name>Nanonull, Inc.</Name>
6      <Desc>
7        <para>The company was established in<b> Vereno</b>in 1995. Nanonull develops
8        nanoelectronic technologies for<i>multi-core processors.</i>February 1999 saw the
9        unveiling of the first prototype <b>Nano-grid.</b>The company hopes to expand its
10       operations <i>offshore</i>to drive down operational costs.
11      </para>
12      <para>White papers and further information will be made available in the near future.
13      </para>
14    </Desc>
15  </Office>
16  <Office>
17    <Name>Nanonull Europe, AG</Name>
18    <Desc>
19      <para>In May 2000, Nanonull<i>Europe</i> was set up in Vienna. The team
20      consists of<b> five research scientists </b>and one administrative staff.</para>
21    </Desc>
22  </Office>
23 </Company-Person>

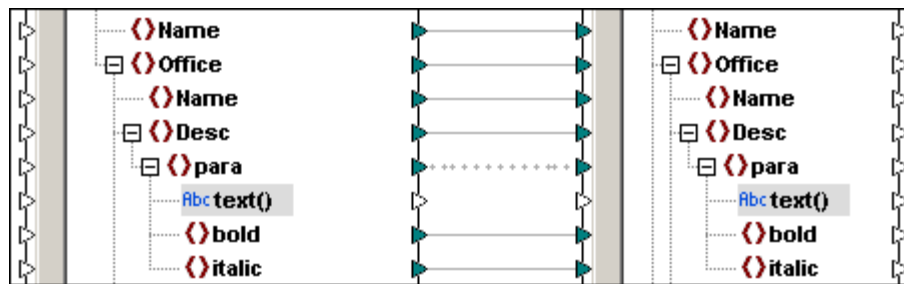
```

Der Mixed Content-Text jeder Beschreibung (Desc) wurde korrekt gemappt; sowohl der Text als auch der Inhalt der Bold- und Italic-Tags wurde so gemappt, wie er in der XML-Quelldatei aufscheint.

6. Wechseln Sie zurück zur Mapping-Ansicht.

Entfernen von Text Nodes aus Mixed Content-Elementen:

1. Klicken Sie auf den **text()** Node-Konnektor und drücken Sie die Entf-Taste um ihn zu löschen.



2. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Ergebnis:

- Alle **Text** Nodes des Elements "para" wurden entfernt.
- Gemappter bold- und italic-Textinhalt wird beibehalten
- Die **Reihenfolge** der Datenelemente "bold" und "italic" entspricht weiterhin der in der XML-Quelldatei!

Mixed Content-Beispiel

Das folgende Beispiel finden Sie im Ordner [...MapForceExamples](#) unter "ShortApplicationInfo.mfd".

Unten sehen Sie ein Fragment der XML-Quelldatei für dieses Beispiel.

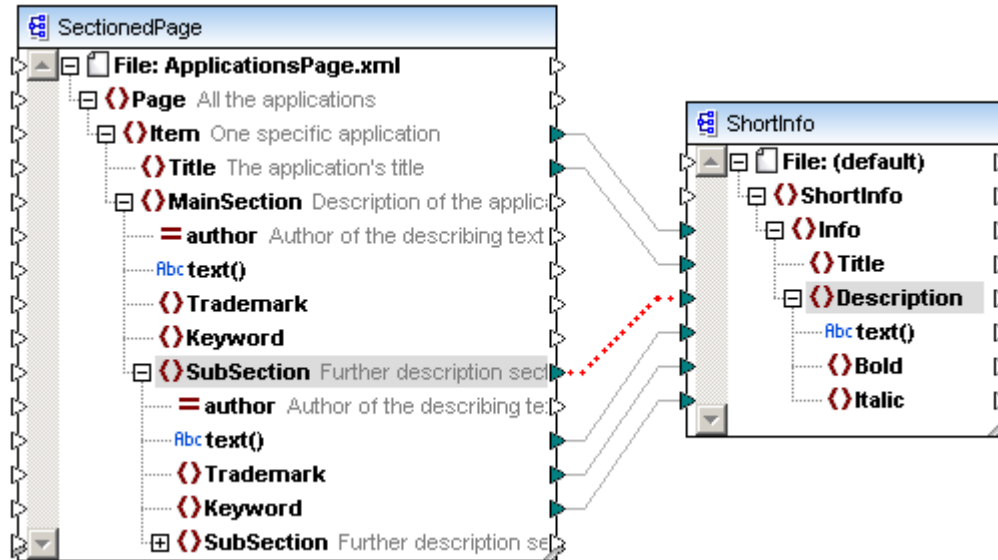
```

<Page xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SectionedPage.xsd">
  <Item>
    <Title>XMLSpy</Title>
    <MainSection author="altova">
      Altova <Trademark>XMLSpy</Trademark>
      <SubSection>Altova <Trademark>XMLSpy</Trademark> 2005 Enter
is the industry standard <Keyword>XML</Keyword> development environment
editing, debugging and transforming all <Keyword>XML</Keyword> technolo
automatically generating runtime code in multiple programming languages
    </MainSection>
  </Item>

```

In der nachfolgenden Abbildung sehen Sie das Mapping. Bitte beachten Sie:

- Der Konnektor für das Datenelement "SubSection" ist vom Typ "Mixed Content" und wird auf das Datenelement "Description" im XML/Schema-Zieldokument gemappt.
- Die text() Nodes werden in der Zielkomponente auf das Datenelement "Bold" gemappt.
- Der Trademark-Text wird auf das Datenelement "Bold" im Zieldokument gemappt.
- Der Keyword-Text wird auf das Datenelement "Italic" im Zieldokument gemappt.



Ergebnis des Mappings:

- Der Mixed Content-Text der einzelnen Beschreibungen wurde korrekt gemappt; sowohl der Text als auch der Inhalt der Tags "Bold" und "Italic" wurde so gemappt, wie er im XML-Quelldokument aufscheint.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <ShortInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="
   C:/PROGRA~1/Altova/MapForce2005/MapForceExamples/ShortInfo.xsd">
3    <Info>
4      <Title>XMLSpy</Title>
5      <Description>Altova <Bold>XMLSpy</Bold> 2005 Enterprise Edition is the industry standard
   <Italic>XML</Italic> development environment for modeling, editing, debugging and transforming
   all <Italic>XML</Italic> technologies, then automatically generating runtime code in multiple
   programming languages.</Description>
6    </Info>

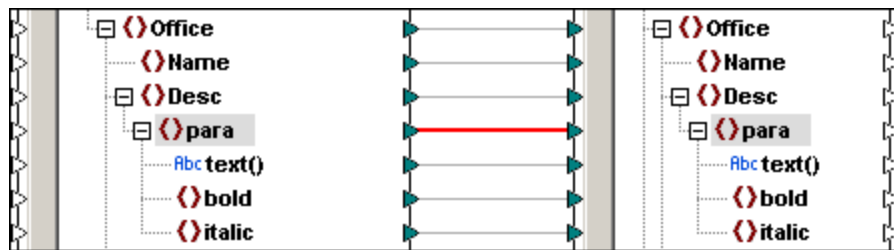
```

Mappen von Mixed Content-Datenelementen mit Hilfe eines quellorientierten Mappings

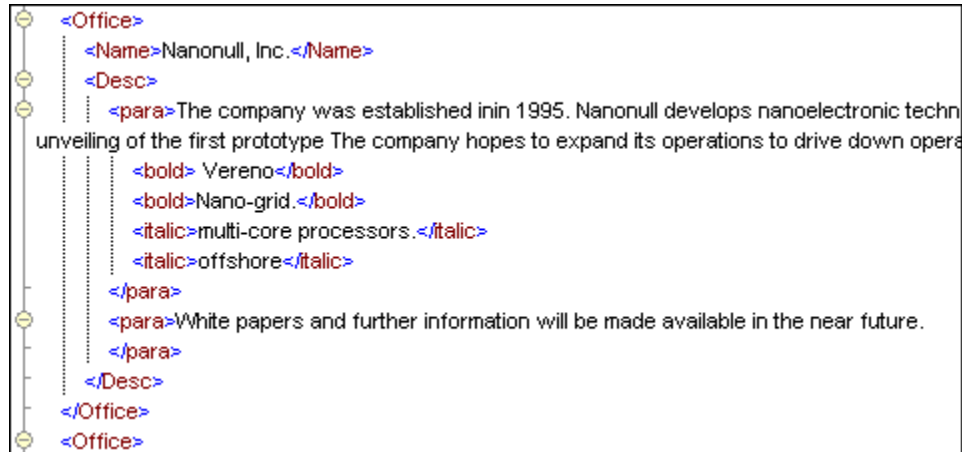
In diesem Abschnitt werden die Ergebnisse beschrieben, die Sie bei Definition eines Standard-Mappings (oder bei Verwendung von Standard-Konnektoren) für **Mixed Content**-Datenelemente erhalten. Sie finden die im folgenden Beispiel (**Tut-OrgChart.mfd**) verwendeten Dateien im Ordner [.../MapForceExamples/Tutorial/](#).

Erstellen von Standardverbindungen zwischen Mixed Content-Datenelementen:

1. Erstellen Sie einen Konnektor zwischen den zwei **para**-Elementen. Daraufhin wird eine Meldung angezeigt, in der Sie gefragt werden, ob MapForce die Konnektoren als quellorientiert erstellen soll.
2. Klicken Sie auf "Nein" um ein Standardmapping zu erstellen.



3.. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Ergebnis:

Ein Mapping von Mixed Content-Datenelementen mit einem Standard-Mapping hat das folgende Ergebnis:

- Text() **Inhalt** wird unterstützt/gemappt
- Die Start-/End-Tags der Child-Nodes, bold und italic werden aus dem text-Node entfernt.
- Die Child Nodes erscheinen nach dem Mixed Content Node text.
- Die **Reihenfolge** der Child Nodes ist von der Reihenfolge der XML/Schema-Zielfdatei abhängig

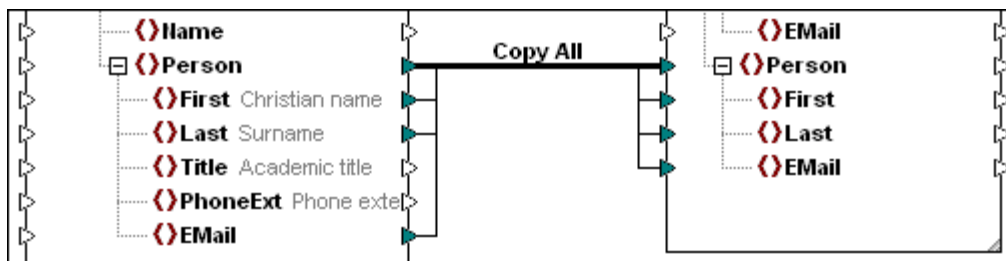
d.h.

Für jedes **para**-Element werden zuerst der text() Node, dann **alle bold**-Datenelemente und anschließend **alle italic**-Datenelemente gemappt. Daraus resultiert die oben gezeigte Reihenfolge: bold, bold - italic, italic. Der Inhalt jedes Datenelements wird gemappt, wenn ein Konnektor vorhanden ist.

6.1.3 "Alles kopieren" Konnektoren

Bei dieser Verbindungsart können Sie automatisch **alle** identischen Datenelemente in Quell- und Zielkomponente miteinander verbinden. Dies bedeutet, dass je nach Quell und Zieltyp alle Quelldatenelemente in die Zielkomponente **kopiert** werden, wenn der Quell- und Zieltyp entweder identisch sind oder der Zieltyp: xs:anyType ist

Wenn der Quell- und Zieltyp **nicht identisch** sind und der Zieltyp nicht vom Typ xs:anyType ist, werden die Quelldaten auf die entsprechenden Datenelemente desselben Namens und derselben Hierarchie gemappt. Wenn die Namen der Zieldatenelemente andere sind, wird kein Mapping zum Zieldatenelement erstellt.



Konnektoren vom Typ "Alles kopieren" werden durch eine einzige fett gedruckte Linie, die die verschiedenen identischen Datenelemente der Quell- und Zielkomponente miteinander verbindet, gekennzeichnet.

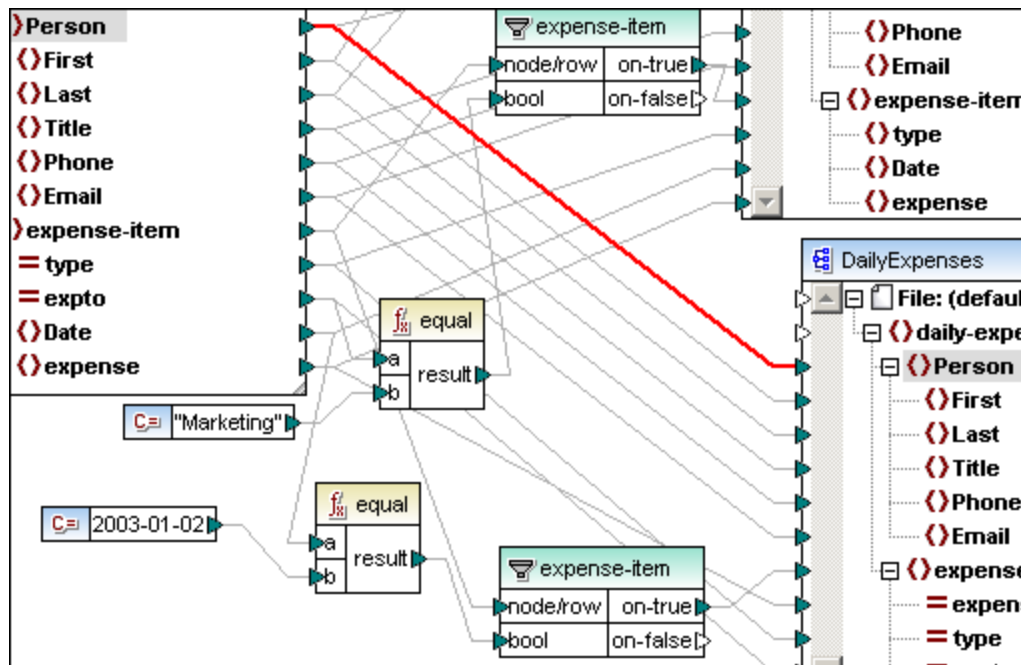
Beachten Sie, dass nur verglichen wird, ob die Namen der Sub-Einträge übereinstimmen, nicht aber ob deren jeweiliger Typ identisch ist.

Derzeit werden die folgenden "Alles kopieren" Verbindungen unterstützt: (i) zwischen complexTypes eines XML Schemas und (ii) zwischen komplexen Komponenten (XML-Schema, Datenbank, EDI) und [komplexen benutzerdefinierten Funktionen/Komponenten](#), die dieselben komplexen Parameter enthalten.

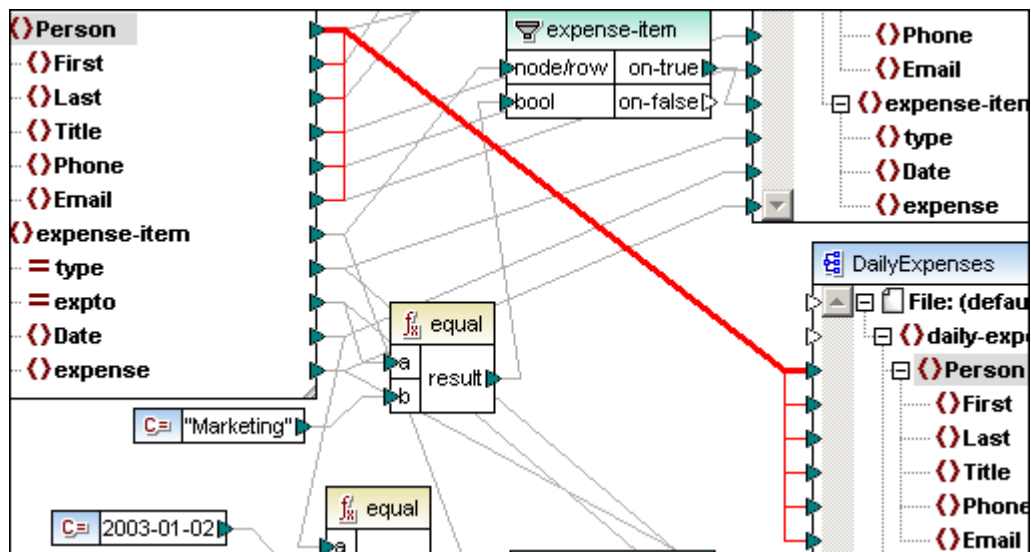
Im Beispiel unten sehen Sie diese Konnektoren in der Datei **MarketingAndDailyExpenses.mfd** aus dem Ordner [...!MapForceExamples](#).

So definieren Sie eine "Alles kopieren"-Verbindung:

1. Rechtsklicken Sie auf einen vorhandenen Konnektor, z.B. auf den "Person" Konnektor und wählen Sie im Kontextmenü die Option "Alles kopieren". Es erscheint eine Meldung, die besagt, dass alle Verbindungen zu Sub-Einträgen der Zielkomponente durch die "Alles kopieren"-Verbindung ersetzt werden.



2. Klicken Sie auf OK, um "Alles kopieren" Konnektoren zu erstellen.



Es werden nun alle Konnektoren zur Zielkomponente und alle Verbindungen zwischen Quell- und Ziel-Datenelementen mit demselben Namen erstellt.

Bitte beachten Sie:

- Wenn die bestehenden Ziel-Verbindungen gelöscht werden, werden Konnektoren von anderen Quellkomponenten oder anderen Funktionen ebenfalls gelöscht.
- Diese Art von Verbindung kann nicht zwischen einem Datenelement und dem Root-Element einer Schema-Komponente erzeugt werden.
- Nachdem Sie diese Methode verwendet haben, können einzelne Konnektoren aus der "Alles kopieren" Gruppe nicht mehr gelöscht oder neu verbunden werden.

Auflösen/Löschen von "Alles kopieren" Konnektoren:

1. Verbinden Sie ein beliebiges Datenelement mit einem Sub-Eintrag der "Alles

kopieren"-Verbindung in der Zielkomponente.

Sie werden darauf aufmerksam gemacht, dass im Zieldatenelement nur ein Konnektor vorhanden sein darf. Klicken Sie auf "Ersetzen", um den Konnektor zu ersetzen.

2. Klicken Sie im nächsten Meldungsfeld auf die Schaltfläche "Alles kopieren-Verbindung auflösen".

Die "Alles kopieren"-Verbindung wird durch einzelne Konnektoren zu den Zielkomponenten ersetzt.

"Alles kopieren" Verbindungen und benutzerdefinierte Funktionen

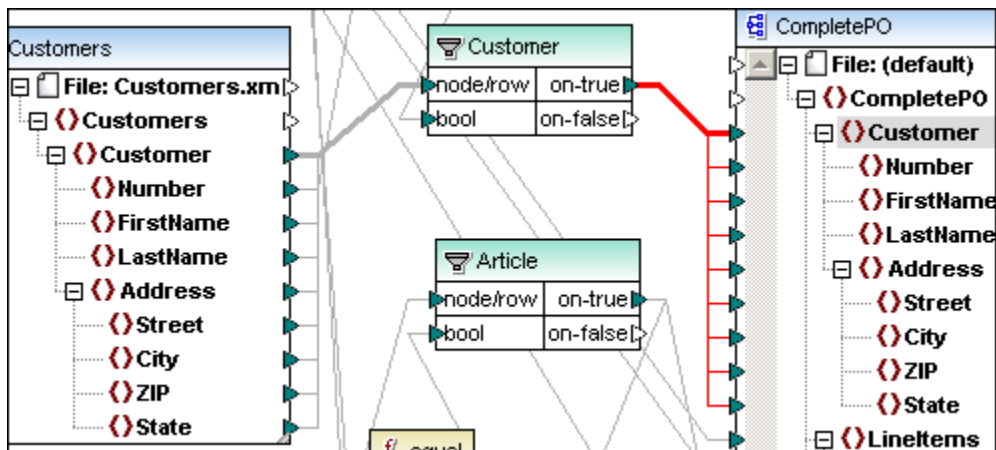
Beim Erstellen von "Alles kopieren" Verbindungen zwischen einem Schema und einem benutzerdefinierten Funktionsparameter müssen die beiden Komponenten auf demselben Schema basieren! Sie müssen jedoch nicht notwendigerweise dasselbe Root-Element haben. Ein Beispiel dazu finden Sie unter [Definieren komplexer Output-Komponenten](#).

"Alles kopieren"-Verbindungen und Filter

"Alles kopieren"-Verbindungen können auch über Filterkomponenten erstellt werden, wenn die Quellkomponente:

- aus strukturierten Daten besteht, d.h. aus einer Schema-, Datenbank- oder EDI-Komponente.
- Daten über einen komplexen Ausgabeparameter einer benutzerdefinierten Funktion oder eines Webservice erhält.
- Daten über eine weitere Filterkomponente erhält.

Nur die gefilterten Daten werden an die Zielkomponente übergeben.

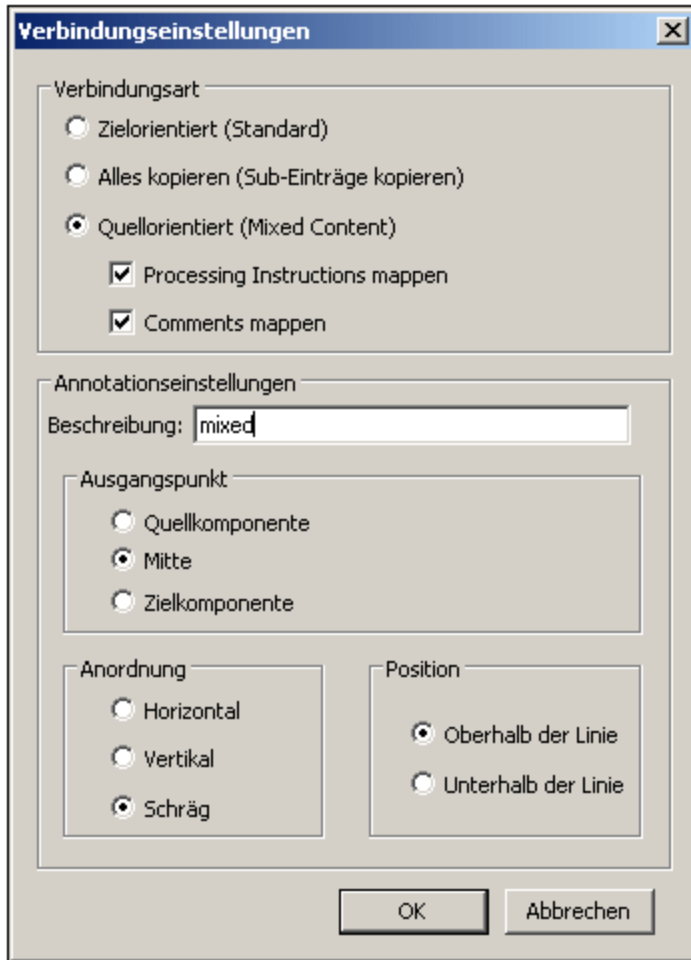


So definieren Sie eine "Alles kopieren"-Verbindung über eine Filterkomponente:

1. Erstellen Sie einen Konnektor vom **on-true/on-false**-Datenelement zum Zieldatenelement, z.B. Customer.
2. Klicken Sie mit der rechten Maustaste auf den Konnektor und wählen Sie im Kontextmenü den Eintrag "Alles kopieren (Sub-Einträge kopieren)". Der "Alles kopieren" Konnektor zwischen Datenelementen desselben Namens wird erstellt.

6.2 Verbindungseinstellungen

Durch Rechtsklick auf einen Konnektor und Auswahl des Befehls **Eigenschaften** aus dem Kontextmenü oder durch Doppelklick auf einen Konnektor wird das Dialogfeld **Verbindungseinstellungen** geöffnet, in welchem Sie die spezifischen Einstellungen (mixed content) für den aktuellen Konnektor festlegen können. Beachten Sie, dass nicht verfügbare Optionen ausgegraut sind.

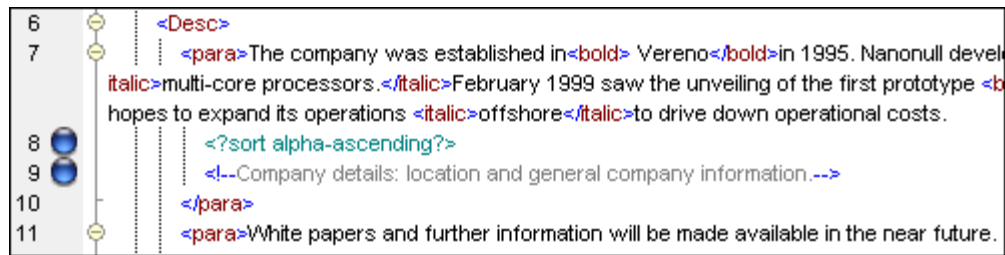


Verbindungstyp

Bei Datenelementen vom Typ **complexType** können Sie für das Mapping eine der folgenden Verbindungsarten auswählen (Bitte beachten Sie, dass diese Einstellungen auch für **complexType**-Datenelemente gelten, die keine Text Nodes enthalten!):

- **Zielorientiert (Standard):** Ändert die Verbindungsart in Standard-Mapping. Nähere Informationen dazu siehe: ["Zielorientiert / Standard-Mapping"](#).
- **Alles kopieren (Sub-Einträge kopieren):** Ändert die Verbindungsart in "Alles kopieren" und verbindet automatisch alle identischen Datenelemente aus der Quell- und Zielkomponente. Nähere Informationen dazu finden Sie unter ["Alles kopieren"-Verbindungen](#).
- **Quellorientiert (Mixed Content):** Ändert die Verbindungsart in quellorientiertes / Mixed Content Mapping, sodass eine Auswahl zusätzlicher Elemente gemappt werden kann. Die zusätzlichen Elemente müssen **Child-Elemente** des gemappten Datenelements in der XML-Quelldatei sein, damit sie gemappt werden können. Wenn Sie die

Kontrollkästchen **Processing Instructions mappen** und/oder **Comments mappen** aktivieren, werden diese Daten in die Ausgabedatei inkludiert.



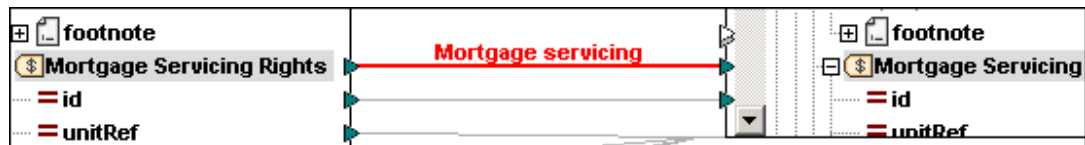
Bitte beachten Sie: CDATA-Abschnitte werden als Text behandelt.

Annotationseinstellungen

Sie können die einzelnen Konnektoren mit einer Beschriftung versehen, um Ihr Mapping ausführlich zu kommentieren. Wenn Sie einen Buchstaben in das Feld "Beschreibung" eingeben, werden auch die Gruppenbereiche "Ausgangspunkt", "Anordnung" und "Position" aktiv und können bearbeitet werden. Diese Option steht für **alle Verbindungstypen** zur Verfügung.

So fügen Sie eine Annotation zu einem Konnektor hinzu:

1. Geben Sie den Namen des gerade ausgewählten Konnektors in das Feld **Beschreibung** ein.
Damit werden alle Optionen in der Gruppe "Annotationseinstellungen" aktiviert.
2. In den restlichen Bereichen können Sie den **Ausgangspunkt, die Ausrichtung und Position** der Beschriftung definieren.
3. Aktivieren Sie in der Symbolleiste die Schaltfläche "Annotationen anzeigen"  um den Annotationstext zu sehen.



Anmerkung: Wenn die Schaltfläche "Annotationen anzeigen" deaktiviert ist, können Sie den Annotationstext dennoch sehen, wenn Sie den Mauszeiger über den Konnektor platzieren. Daraufhin wird der Text in einem Popup-Fenster angezeigt, wenn die Schaltfläche **Tipps anzeigen**  in der Symbolleiste "Ansichtsoptionen" aktiv ist.

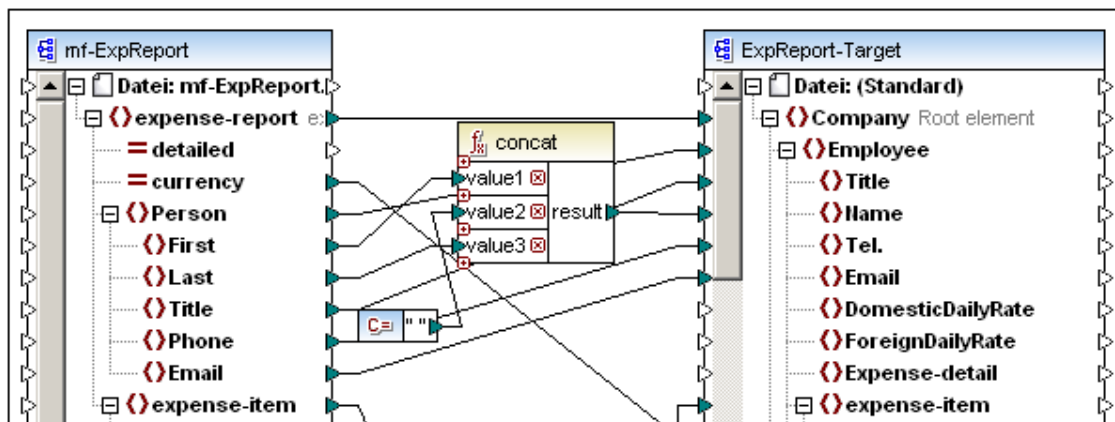
6.3 Verschieben von Konnektoren

Verschieben von Konnektoren und die Auswirkung auf Child-Konnektoren

Wenn Sie einen Parent-Konnektor auf einen anderen Parent-Konnektor verschieben, findet MapForce automatisch die identen Child-Verbindungen unter dem neuen Konnektor. Diese Funktion ist nicht mit der Option "Identische Sub-Einträge automatisch verbinden" identisch, da das Ergebnis mit Hilfe anderer Regeln erzielt wird.

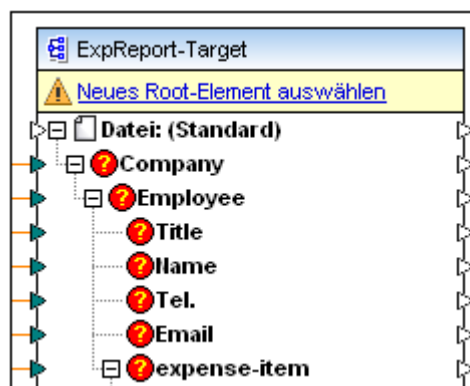
Häufig wird diese Funktion verwendet, wenn Sie das Root-Element des Zielschemas in einem bestehenden Mapping ändern. Normalerweise müssten Sie in diesem Fall alle untergeordneten Konnektoren manuell mappen.

In diesem Beispiel wird die Datei **Tut-ExpReport.mfd** aus dem Ordner ...\\MapForceExamples\\Tutorial verwendet.

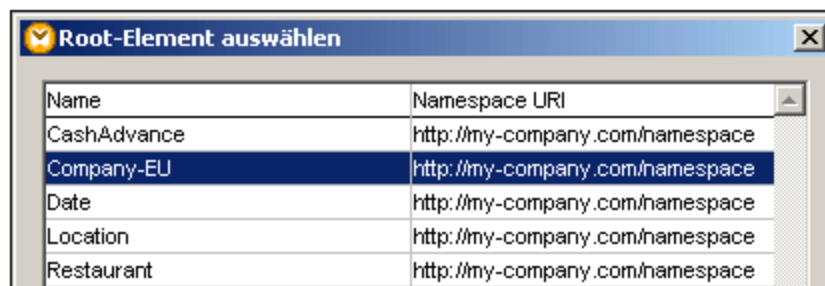


Wenn das Company Root-Element des Zielschemas in "Company-EU" geändert wird, wird in MapForce die Meldung "Geänderte Dateien" angezeigt.

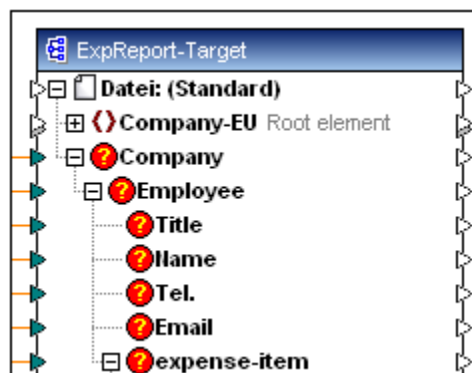
1. Klicken Sie auf die Schaltfläche **Neu laden**, um das aktualisierte Schema neu zu laden. Sie sehen nun, dass mehrere Nodes fehlen, da sich das Root-Element geändert hat.



2. Klicken Sie oben in der Komponente auf den Link "Neues Root-Element auswählen".

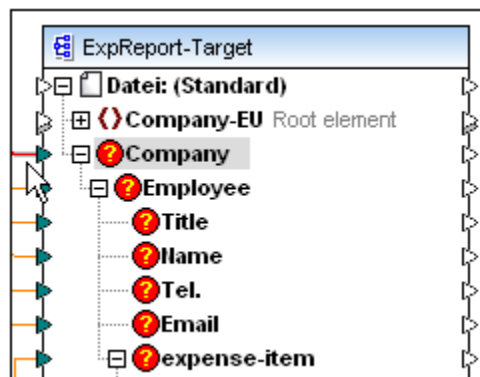


3. Wählen Sie das aktualisierte Root-Element Company-EU aus und klicken Sie zur Bestätigung auf OK.

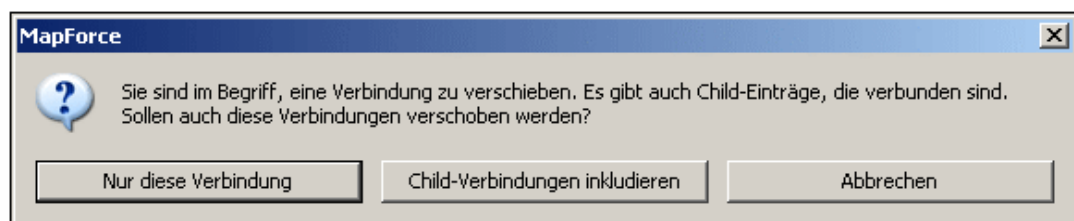


Das Root-Element "Company-EU" wird nun am oberen Rand der Komponente angezeigt.

4. Klicken Sie auf den Konnektor des Datenelements "Company" und ziehen Sie ihn mit der Maus auf das neue Root-Element "Company-EU".

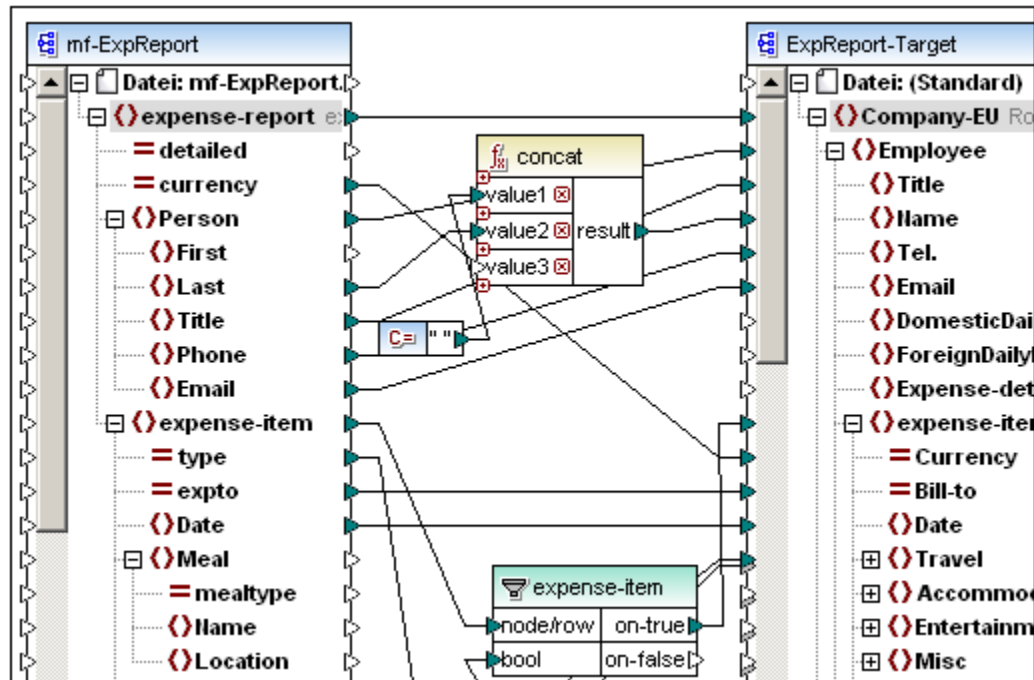


Daraufhin erscheint eine Meldung, in der Sie gefragt werden, welche Konnektoren verschoben werden sollen.



5. Klicken Sie auf die Schaltfläche "Child-Verbindungen inkludieren", wenn die

Child-Konnektoren gemappt werden sollen.
Die Nodes für die "fehlenden Datenelemente" wurden entfernt und alle Konnektoren wurden auf die richtigen Child-Elemente unter dem neuen Root-Element gemappt.



Bitte beachten Sie:

Wenn der Ziel-Node/das Zieldatenelement denselben Namen wie der Quell-Node hat, sich aber in einem anderen Namespace befindet, so enthält die Meldung eine zusätzliche Schaltfläche "Child-Verbindungen inkludieren und Namespace mappen".

Wenn Sie auf diese Schaltfläche klicken, werden die Child-Konnektoren desselben Namespace wie die des Parent-Node der Quellkomponente auf dieselben Child-Nodes unter dem Node des anderen Namespace verschoben. Wenn sich die Parent-Nodes also nur im Namespace unterscheiden, dann dürfen sich auch die Child-Nodes in dieser Hinsicht voneinander unterscheiden und können automatisch gemappt werden.

Sie können das Root-Element auch ändern, indem Sie auf die Komponentenüberschrift klicken und im Kontextmenü den Befehl "Root-Element ändern" auswählen.

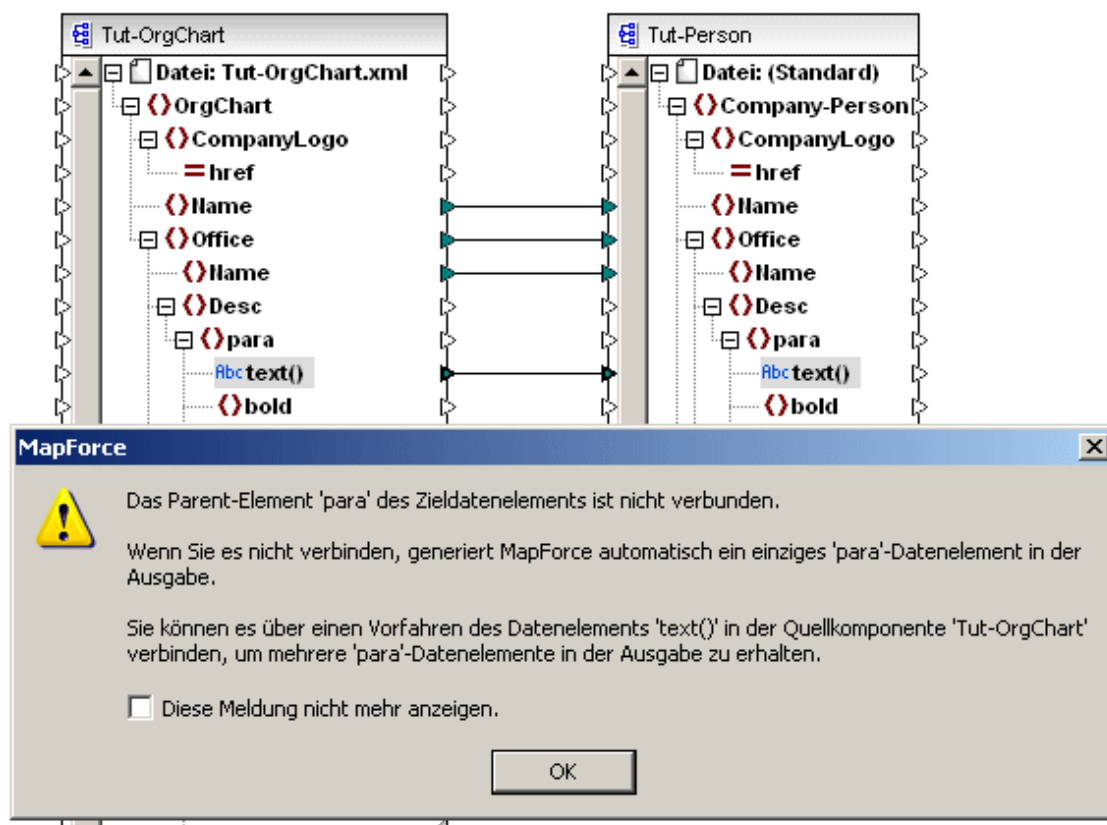
6.4 Verbindungen und Mapping-Ergebnisse

Wenn Sie manuell eine Verbindung zwischen einem Quell- und einem Zieldatenelement erstellen, analysiert MapForce automatisch die möglichen Ergebnisse. Wenn Sie zwei Subelemente mappen, wird eine Meldung angezeigt, in der vorgeschlagen wird, auch das übergeordnete Quelldatenelement mit dem übergeordneten Zieldatenelement zu verbinden.

Dadurch verhindern Sie, dass in der Mapping-Vorschau nur ein einziges Subelement im Ausgabefenster angezeigt wird. Im Allgemeinen ist dies der Fall, wenn der Quell-Node eine Sequenz anstatt eines einzelnen Werts liefert.

Das unten gezeigte Mapping Tut-OrgChart.mfd steht im Ordner ...\\MapForceExamples\\Tutorial zur Verfügung.

Wenn Sie das Quelldatenelement text() mit dem Zieldatenelement text() verbinden, erscheint ein Meldungsfeld, in dem Sie darauf aufmerksam gemacht werden, dass das übergeordnete Zieldatenelement "para" nicht verbunden ist und nur einmal in der Ausgabe generiert wird. Um mehrere para-Elemente in der Zielkomponente zu generieren, verbinden Sie die das para-Element in der Quellkomponente mit dem para-Element in der Zielkomponente.

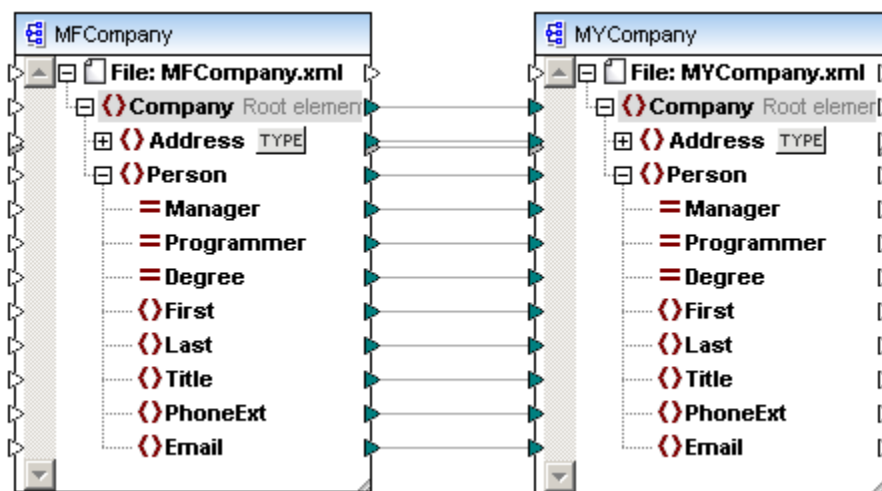


6.5 Fehlende Datenelemente

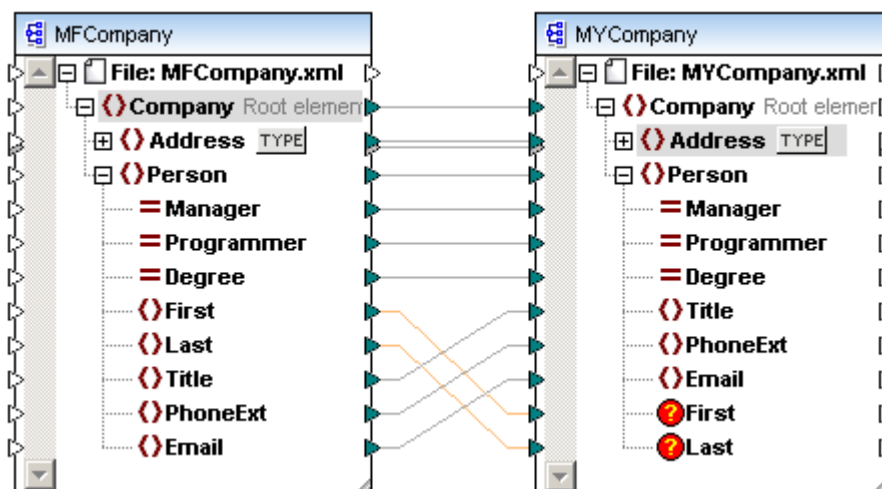
Oft kommt es vor, dass sich die Struktur einer der Komponenten in einem Mapping ändert, dass z.B. Attribute oder Elemente zu einem Schema hinzugefügt oder daraus gelöscht werden. In MapForce werden nun Platzhalter-Datenelemente verwendet, um alle Konnektoren sowie alle relevanten Verbindungsdaten zwischen Komponenten beizubehalten, wenn Datenelemente gelöscht wurden.

Beispiel:

In diesem Beispiel wird die Schemadatei **MFCompany.xsd** verwendet. Das Schema wird in **MyCompany.xsd** umbenannt und es wird ein Konnektor zwischen den beiden Company-Datenelementen in den beiden Schemas angelegt. Daraufhin werden Konnektoren für alle Child-Datenelemente zwischen den Komponenten erstellt, wenn der Befehl "Identische Sub-Einträge automatisch verbinden" aktiv ist.



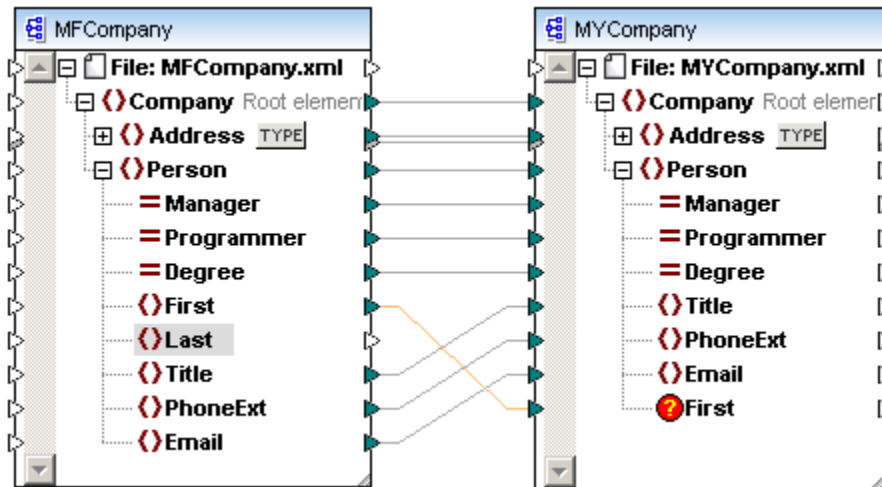
Während der Bearbeitung von **MyCompany.xsd** in XMLSpy werden die Elemente "First" und "Last" im Schema gelöscht. Wenn Sie wieder zu MapForce wechseln, wird ein Dialogfeld angezeigt, dass Dateien geändert wurden und Sie werden aufgefordert, das Schema neu zu laden. Wenn Sie auf "Neu laden" klicken, werden die Komponenten in MapForce aktualisiert.



Die gelöschten **Datenelemente** und ihre **Konnektoren** erscheinen in der Komponente "MyCompany" nun markiert. Sie können die Konnektoren nun gegebenenfalls mit anderen Datenelementen verbinden oder die Konnektoren löschen.

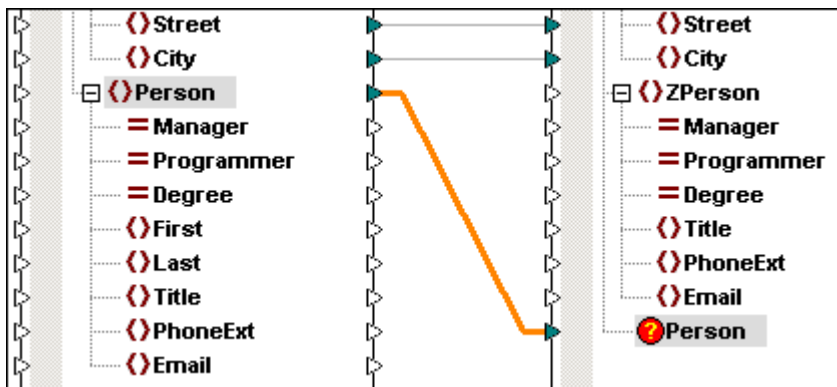
Beachten Sie, dass Sie immer noch eine Vorschau des Mappings anzeigen (oder Code generieren) können, dass aber im Fenster "Meldungen" Warnmeldungen angezeigt werden, falls Sie dies zu diesem Zeitpunkt tun. Alle Verbindungen zu oder von fehlenden Datenelementen werden bei der Vorschau oder Codegenerierung ignoriert.

Wenn Sie auf einen der markierten Konnektoren klicken und ihn löschen, wird das "fehlende" Datenelement - z.B. "Last" in "MyCompany" aus der Komponente gelöscht.



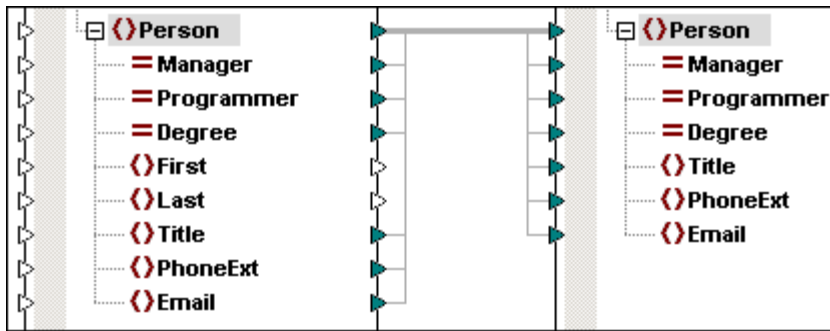
Umbenannte Datenelemente:

Wenn ein Parent-Datenelement umbenannt wird, z.B. Person in ZPerson, so wird der Original-Parent-Datenelement-Konnektor beibehalten und die Child-Elemente und deren Konnektoren werden gelöscht.



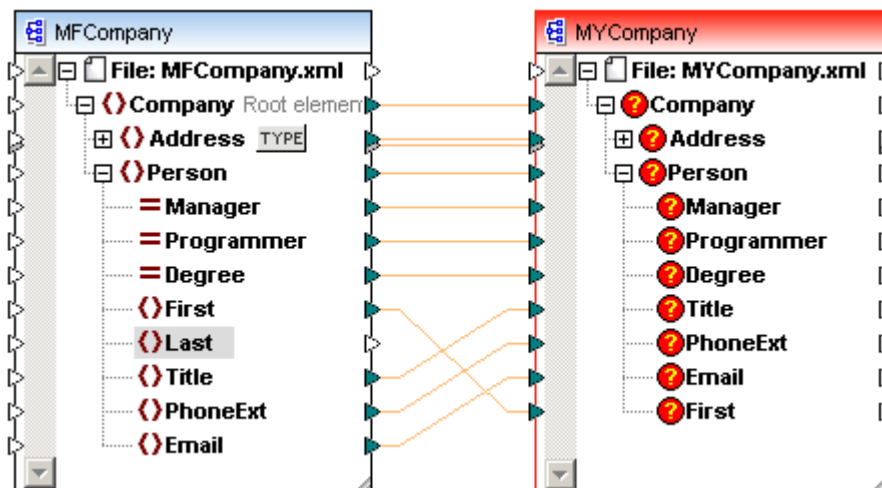
"Alles kopieren"-Konnektoren und fehlende Datenelemente:

"Alles kopieren"-Verbindungen werden genau wie normale Verbindungen behandelt, wobei der einzige Unterschied darin besteht, dass die Verbindungen zu den fehlenden Subeinträgen nicht beibehalten oder angezeigt werden.



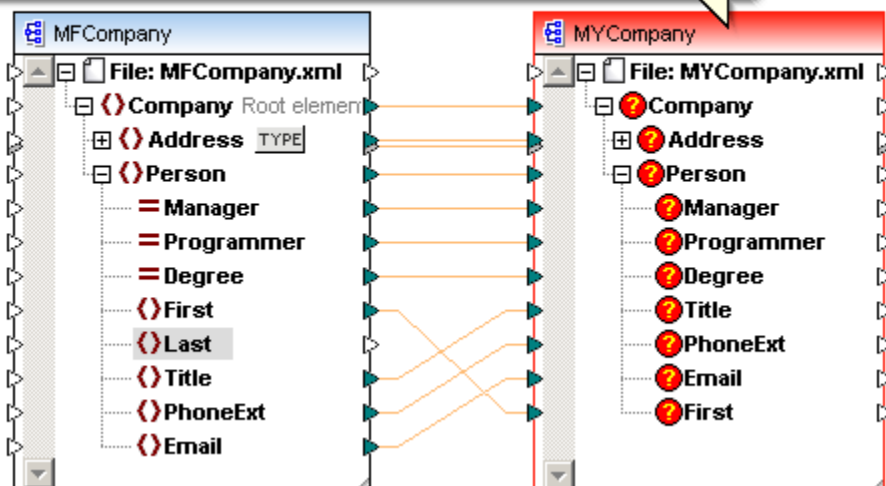
Umbenannte oder gelöschte Komponentenquellen:

Wenn die **Datenquelle** einer Komponente, d.h. ein Schema, eine Datenbank usw.. umbenannt oder gelöscht wurde, werden alle darin enthaltenen Datenelemente markiert. Der rote Rahmen um die Komponente zeigt an, dass es keine gültige Verbindung zu einem Schema oder einer Datenbank-Datei gibt. Es kann keine Vorschau und kein Code generiert werden.



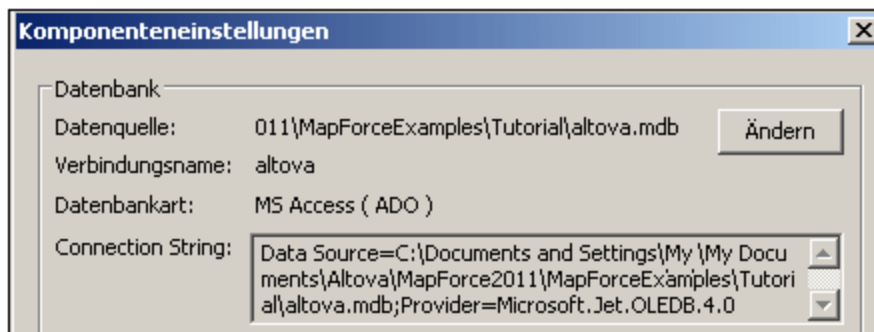
Wenn Sie den Mauszeiger über die markierte Komponente platzieren, wird ein Popup-Fenster mit den dazugehörigen Informationen angezeigt.

This component does not have any valid structure information.
Local file 'C:\2010\MapForceExamples\Tutorial\MYCompany.xsd' was not found.



Wenn Sie auf die Titelleiste der markierten Komponente doppelklicken, wird das Dialogfeld "Komponenteneinstellungen" geöffnet. Wenn Sie auf die Durchsuchen-Schaltfläche in der Gruppe **Schema-Datei** klicken, können Sie eine andere Version oder eine Sicherungskopie des Schemas öffnen. Nähere Informationen dazu finden Sie unter "[Komponente](#)" in Abschnitt "Referenz".

Wenn Sie in dem Dialogfeld, das geöffnet wird, wenn es sich bei der Komponente um eine Datenbank handelt, auf die Schaltfläche **Ändern** klicken, können Sie eine andere Datenbank auswählen oder ändern, welche Tabellen in der Datenbankkomponente angezeigt werden. Konnektoren zu Tabellen desselben Namens werden beibehalten.



Alle gültigen/korrekten Verbindungen (und relevanten Datenbankdaten, wenn es sich bei der Komponente um eine Datenbank handelt) werden beibehalten, wenn Sie ein Schema oder eine Datenbank derselben Struktur auswählen.

6.6 Verkettete Mappings / Weiterleitungskomponenten

MapForce unterstützt Mappings, die aus mehreren Komponenten in einer Mapping-Kette bestehen. Verkettete Mappings sind Mappings, in denen mindestens eine Komponente gleichzeitig als Quell- und Zielkomponente fungiert. Mit Hilfe einer solchen Komponente wird eine Ausgabe erzeugt, die später als Input für den nächsten Mapping-Schritt in der Kette verwendet wird. Eine solche Komponente wird als "Zwischenkomponente" bezeichnet.

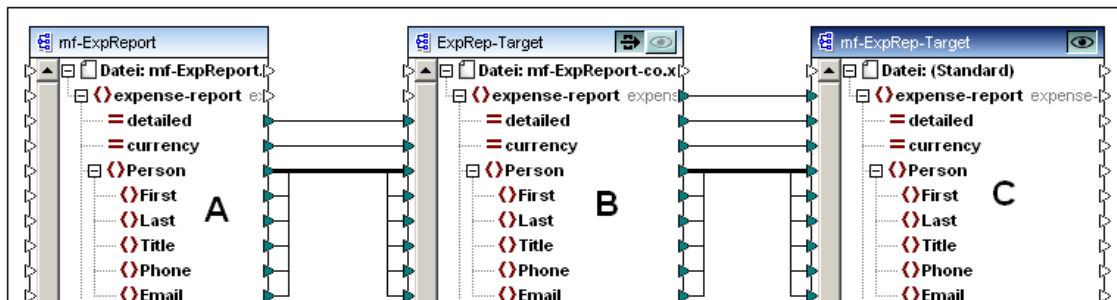
In verketteten Mappings wird eine Funktion, namens "Weiterleitung" verwendet. Mit Hilfe dieser Funktion können Zwischenresultate in Form von "Zwischenkomponenten" für die Vorschau, die Ausführung über die Befehlszeile und die Codegenerierung erzeugt werden. Die Weiterleitungsfunktion ist eine Vorschaufunktion, mit der Sie die verschiedenen Phasen eines verketteten Mappings im Fenster "Ausgabe" anzeigen können. Für die Anzeige der Vorschau wird der Built-in-Ausführungsprozessor verwendet. Die Ergebnisse werden in Form von temporären Dateien erzeugt.

Bei Ausführung des Mappings über die Befehlszeile bzw. bei der Generierung von Code, wird unabhängig davon, was in die Felder "XML-Input-Datei" und "XML-Output-Datei" der Zwischenkomponente eingegeben wurde, die gesamte Mapping-Kette ausgeführt und die Ausgabedaten eines vorherigen Mapping-Schritts werden als Eingabedaten für den nächsten Mapping-Schritt verwendet.

Anmerkung:

Die Funktion "Weiterleitung" steht nur für dateibasierte "Zwischenkomponenten", also XML-, CSV-, TXT-Dateien usw. zur Verfügung. Datenbankkomponenten können als Zwischenkomponenten verwendet werden, doch wird die Schaltfläche "Weiterleitung" nicht angezeigt. Die Zwischenkomponente wird für die Ausgabevorschau oder die Codegenerierung immer von Grund auf neu generiert. Dies wäre mit einer Datenbank nicht möglich, da sie vor jeder Neugenerierung gelöscht werden müsste.

In der Abbildung unten sehen Sie als Beispiel die drei Komponenten A, B und C, wobei C die Zielkomponente ist. Die Komponente B (ExpRep-Target) ist die "Zwischenkomponente", da sie sowohl Input- als auch Output-Verbindungen aufweist.



Bei der Ausführung eines verketteten Mappings über die Befehlszeile oder bei Ausführung des generierten Codes führt das Mapping alle Schritte in der richtigen Reihenfolge aus und generiert die erforderlichen Ausgabedateien.



Schaltfläche "Vorschau"

Sowohl die Komponente B als auch die Komponente C verfügt über eine Vorschau-Schaltfläche. Auf diese Art können Sie sowohl das Mapping-Zwischenergebnis von B als auch das endgültige Ergebnis des verketteten Mappings der Komponente C im Built-in-Ausführungsprozessor anzeigen. Klicken Sie auf die Vorschau-Schaltfläche der jeweiligen Komponente und anschließend auf "Ausgabe", um das Mapping-Ergebnis zu sehen.

"Zwischenkomponenten", bei denen die Schaltfläche "Weiterleitung" aktiv ist, können nicht in der Vorschau angezeigt werden, da die Vorschau-Schaltfläche automatisch deaktiviert wird. Um die Ausgabe einer solchen Komponente zu sehen, klicken Sie auf die Schaltfläche "Weiterleitung" um sie zu deaktivieren und klicken Sie anschließend auf die Vorschau-Schaltfläche der Zwischenkomponente.



Schaltfläche "Weiterleitung"

Die Zwischenkomponente B verfügt in der Titelleiste der Komponente über eine zusätzliche Schaltfläche mit dem Namen "Weiterleitung".

Wenn die Schaltfläche "Weiterleitung" **aktiv** ist , werden werden alle Daten auf einmal - von Komponente A auf Komponente B und weiter auf Komponente C - in das Vorschaufenster gemappt. Es werden zwei separate Datengruppen generiert:

- das Ergebnis des Mappings von Komponente A auf die Zwischenkomponente B
- das Ergebnis des Mappings der Zwischenkomponente B auf die Zielkomponente C.

Wenn die Schaltfläche "Weiterleitung" **deaktiviert**  ist, wird nur ein Teil der kompletten Mapping-Kette ausgeführt. Welche Daten generiert werden, hängt davon ab, welche Vorschau-Schaltflächen (Komponente B oder C) aktiv sind:

- Wenn die Schaltfläche "Vorschau" der Komponente B aktiv ist, so wird das Ergebnis des Mappings von Komponente A auf Komponente B generiert. Die Mapping-Kette endet somit bei Komponente B. Die Komponente C wird in der Vorschau gänzlich ignoriert.
- Wenn die Schaltfläche "Vorschau" der Komponente C aktiv ist, so wird das Ergebnis des Mappings von Komponente B auf Komponente C generiert. Wenn die Schaltfläche "Weiterleitung" deaktiviert ist, wurde die automatische Verkettung zur Komponente B unterbrochen. Nur der rechte Teil des Mappings wird ausgeführt. Komponente A wird nicht verwendet.

Beachten Sie Folgendes: Wenn das Mapping über die Befehlszeile oder anhand von generiertem Code ausgeführt wird, so wird, unabhängig von den Einstellungen der Schaltfläche "Weiterleitung" der Komponente B sowie von der gerade ausgewählten Vorschaukomponente, die Ausgabe aller Komponenten generiert.

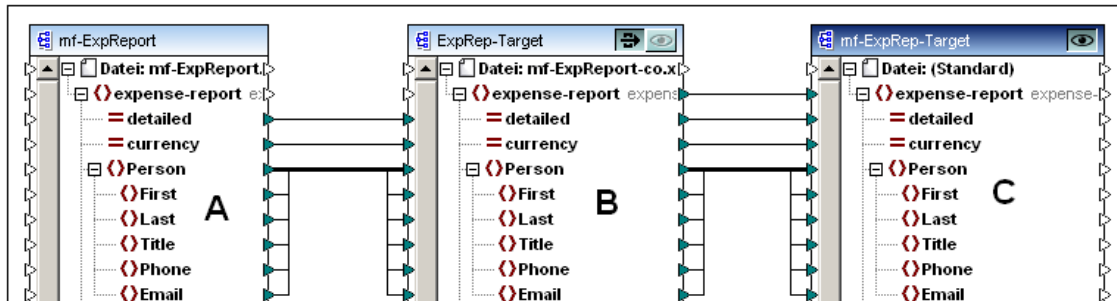
In unserem Beispiel werden zwei Ergebnisdateien generiert! Dies liegt daran, dass MapForce automatisch die Abhängigkeit aller Komponenten analysiert und alle Ausgabedateien der Zwischen- und der Endkomponente in der richtigen Reihenfolge generiert.

Da die Einstellung "Weiterleitung" derzeit nicht aktiv ist, müssen die Dateinamen in den Feldern "XML-Input-Datei" und "XML-Output-Datei" der Zwischenkomponente unbedingt identisch sein.

Nähere Informationen zu diesem Beispiel und eine Beschreibung dazu, welche Unterschiede bei der Übertragung der Quelldaten bestehen, wenn die Schaltfläche "Weiterleitung" **aktiv** bzw. **inaktiv** ist, finden Sie in den folgenden Abschnitten. Ein Beispiel dazu finden Sie unter "[Beispiel für ein verkettetes Mapping](#)".

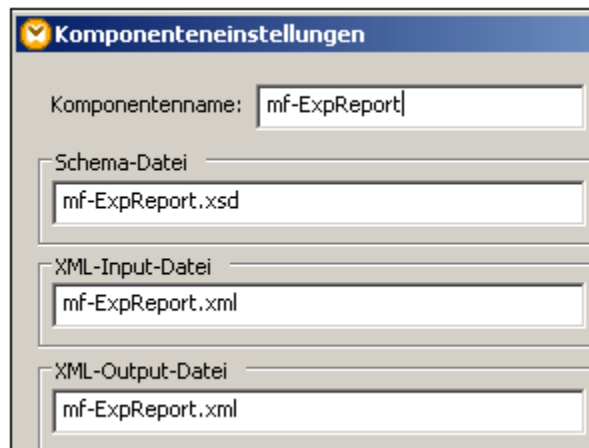
6.6.1 Verkettete Mappings - Weiterleitung aktiv

Die im folgenden Beispiel (**Tut-ExpReport-chain.mfd**) verwendeten Dateien stehen im Ordner [...\\MapForceExamples\\Tutorial\\](#) zur Verfügung.



Das Beispiel **Tut-ExpReport-chain.mfd** ist folgendermaßen eingerichtet:

- Die **Komponente A** liefert anhand einer XML-Beispieldatei alle Mapping-Daten. Die XML-Datei (mf-ExpReport.xml) wird im Dialogfeld "**Komponenteneinstellungen**" im Feld "**XML-Input-Datei**" angezeigt. Die XML-Output-Datei desselben Namens wird automatisch eingefügt, wenn Sie eine XML-Input-Datei definieren.



- Zwischenkomponente B** "Weiterleitung" ist aktiv: Wenn die Weiterleitung aktiviert ist, wird das Feld "**XML-Input-Datei**" der Zwischenkomponente automatisch deaktiviert. Es wird kein Dateiname benötigt, damit das Mapping ausgeführt werden kann, da die Zwischendaten in temporären Dateien gespeichert werden.

Wenn keine XML-Output-Datei definiert wurde, wird ein Standarddateiname verwendet. Wenn im Feld "**XML-Output-Datei**" ein Dateiname definiert wurde, so wird dieser als Dateiname der Ausgabedatei der Zwischenkomponente verwendet.

Beachten Sie, dass Zwischenkomponenten auch dynamische Dateinamen, d.h. Konnektoren zum Eintrag "Datei:" einer Komponente (oder sogar Platzhalterzeichen für Dateinamen) haben können. Nähere Informationen dazu finden Sie unter [Dynamische Input/Output-Dateien pro Komponente](#).

Komponenteneinstellungen

Komponentenname: ExpRep-Target

Schema-Datei: ExpRep-Target.xsd

XML-Input-Datei:

XML-Output-Datei:

- Der Endkomponente C ist **keine** XML-Output-Datei zugewiesen. Die Vorschau-Schaltfläche der Komponente C ist aktiv.

Klicken Sie auf die Schaltfläche "Ausgabe", um eine Vorschau auf die Ergebnisse des Built-in-Ausführungsprozessors zu sehen.

Vorschau 1:

Das Ergebnis des Mappings der Komponente A über die Zwischenkomponente B auf die Zielkomponente C. Dabei handelt es sich um alle Reisekosten unter 1500.

Vorschau von 2 von 1 (2) ExpRep-Target: C:\MapForceExamples\Tutorial\ExpRep-Target.xml

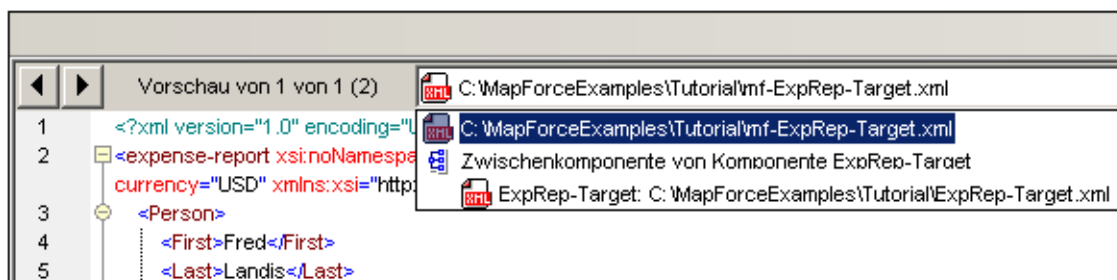
```

<?xml version="1.0" encoding="UTF-8"?>
<expense-report xsi:noNamespaceSchemaLocation="C:/Altova/MapForce2011/MapForceExamples/Tutori
currency="USD" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Person>
    <First>Fred</First>
    <Last>Landis</Last>
    <Title>Project Manager</Title>
    <Phone>123-456-78</Phone>
    <Email>f.landis@nanonull.com</Email>
  </Person>
  <expense-item type="Travel" expto="Development">
    <Date>2003-01-02</Date>
    <Travel Trav-cost="337.88">
      <Destination/>
    </Travel>
    <description>Biz jet</description>
  </expense-item>
  <expense-item type="Travel" expto="Accounting">
    <Date>2003-07-07</Date>
    <Travel Trav-cost="1014.22">
      <Destination/>
    </Travel>
    <description>Ambassador class</description>
  </expense-item>
  <expense-item type="Travel" expto="Marketing">
    <Date>2003-02-02</Date>
    <Travel Trav-cost="2000">
      <Destination/>
    </Travel>
    <description>Hong Kong</description>
  </expense-item>
</expense-report>

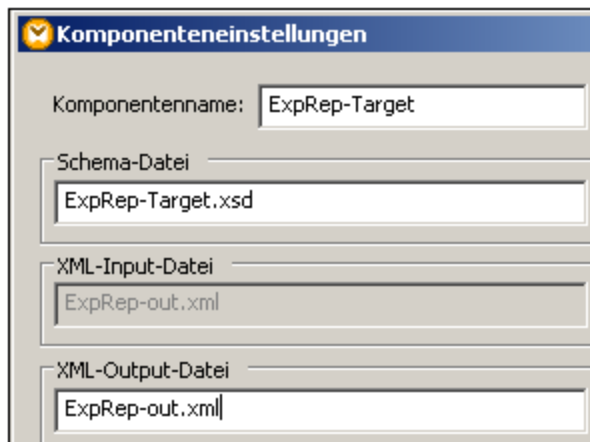
```

Bitte beachten Sie:

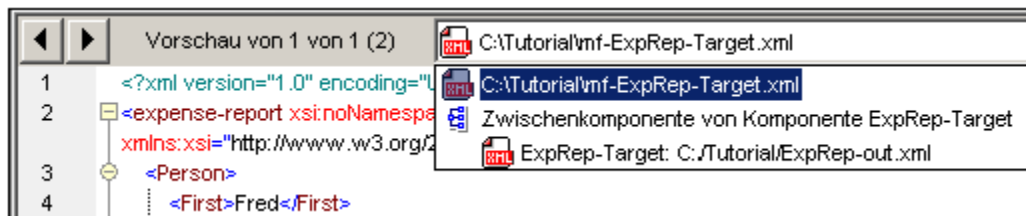
Jedes Mapping-Ergebnis wird in einem eigenen Vorschauenfenster angezeigt. Klicken Sie auf die Pfeilschaltfläche(n) um die nächste/vorherige Vorschau zu sehen.



Wenn Sie auf die Dateiauswahlliste klicken, werden die Ergebnisdateien in einer Hierarchie angezeigt. Ganz oben wird das endgültige Ergebnis der Zielkomponente angezeigt, unterhalb davon die Zwischenergebnisdateien. Klicken Sie auf den Namen einer Datei, um diese auszuwählen oder geben Sie den Namen über die Tastatur ein, um durch die Dateiliste zu navigieren, und drücken Sie die Eingabetaste.



Wenn Sie in der Zwischenkomponente B eine XML-Output-Datei mit dem Namen "ExpRep-out.xml" einstellen, so werden die Daten der Zwischenkomponente B in einer Datei dieses Namens, also ExpRep-out.xml, gespeichert.



Wenn die Schaltfläche "Weiterleitung" aktiviert ist, werden Dateien, die mittels einer Zwischenkomponente erstellt werden, **automatisch** als Temp-Dateien gespeichert und für die weitere Verarbeitung der Ausgabe dieser Komponente verwendet.

Mit der Einstellung "Direkt in die endgültigen Output-Dateien schreiben" (Extras | Optionen | Allgemein) wird festgelegt, ob Zwischendateien als temporäre Dateien oder als physische Dateien gespeichert werden sollen. Für Zwischenkomponenten wird zum Speichern des Zwischenergebnisses ein Standarddateiname verwendet, außer es wird ein dynamischer Dateiname bereitgestellt bzw. durch ein Mapping geliefert.

Vorschau **von 1** gibt die Anzahl der endgültigen Zieldateien an, die von der ausgewählten Zielkomponente aus erzeugt werden können, in diesem Fall 1. Vorschau ... **(2)** bezieht sich auf die Gesamtanzahl der Ergebnisse einschließlich aller Zwischenkomponenten.

Anzeige des Ergebnisses mit Hilfe von StyleVision

Wenn einer Zielkomponente eine SPS-Datei zugewiesen wurde, so werden die Ergebnisdaten bei Klick auf das Register HTML, RTF, PDF oder Word auf dem entsprechenden StyleVision-Register in MapForce angezeigt.



Personal Expense Report

Currency: ☒ Dollars ☐ Euros ☐ Yen Currency \$

☒ Detailed report

Employee Information

Fred

Landis

First Name
Last Name

Project Manager

Title

f.landis@nanonull.com

E-Mail

123-456-78

Phone

Expense List

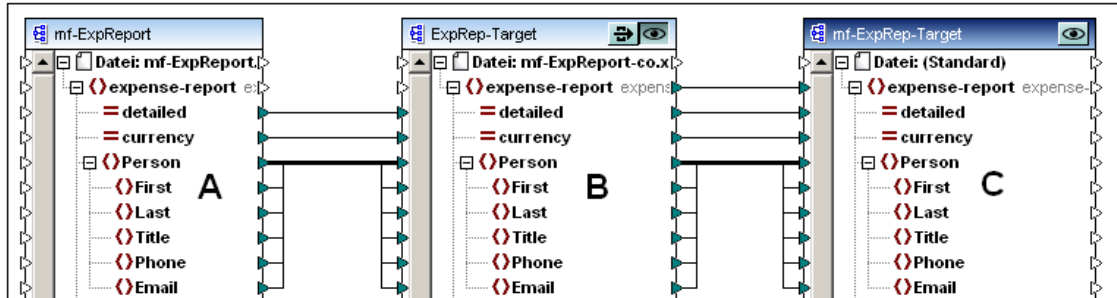
Type	Expense To	Date (yyyy-mm-dd)	Expenses \$		Description
Travel	Development	2003-01-02	Travel 337.88	Lodging	Biz jet
Travel	Accounting	2003-07-07	Travel 1014.22	Lodging	Ambassador class

Mapping
Database Query
Output
 HTML
 RTF

Beachten Sie, dass nur Ausgabedateien von Endkomponenten einer Mapping-Kette auf dem Register "StyleVision" von MapForce angezeigt werden. StyleVision-Ausgabedateien von Zwischenkomponenten können nicht angezeigt werden.

6.6.2 Verkettete Mappings - Weiterleitung deaktiviert

Das Beispiel **Tut-ExpReport-chain.mfd** funktioniert anders, wenn die Schaltfläche "Weiterleitung" für die Komponente B **deaktiviert** ist.



Die automatische Weiterleitung der Daten von Komponente A über Komponente B an Komponente C wurde durch Deaktivierung der Schaltfläche "Weiterleitung" unterbrochen. Die Vorschau-Schaltflächen der Komponenten B und C bestimmen, welcher Teil der Mapping-Kette generiert wird.

MapForce generiert die Ausgabe für die Komponente, in der die Vorschau-Schaltfläche aktiv ist.

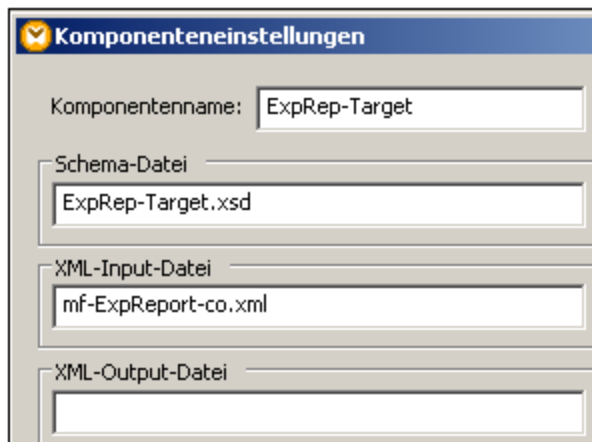
- Wenn die Vorschau-Schaltfläche der Komponente **B** aktiv ist, so wird das Ergebnis des Mappings der Komponente A auf die Komponente B generiert. Komponente C wird ignoriert.

Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie eine Vorschau auf die Ergebnisse im Built-in-Ausführungsprozessor.

Vorschau:

Das Ergebnis des Mappings von Komponente A auf die Zwischenkomponente B, d.h. alle Reisekostenposten.

- Wenn die Vorschau-Schaltfläche der Komponente C aktiv ist, werden die Daten von der Zwischenkomponente B auf Komponente C gemappt. Komponente A wird ignoriert. Der Komponente B wurde als XML-Input-Datei die Datei mf-ExpReport-co.xml zugewiesen. Informationen zum [Speichern des Zwischenergebnisses eines Mappings](#) finden Sie weiter unten.



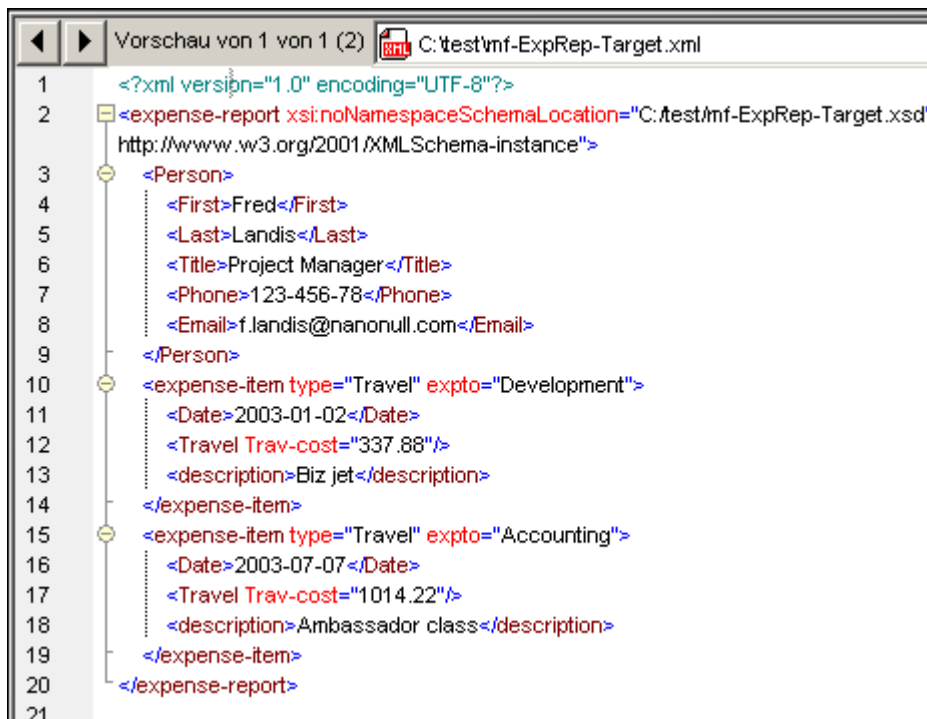
MapForce öffnet die Zwischendatei und mappt Ihre Daten auf Komponente C. Wenn die Input-Datei von Komponente B existiert, werden mit diesem Mapping-Ausgabedaten erzeugt. Es muss eine Datei im Feld "XML-Input-Datei" eingetragen sein, damit das Mapping ausgeführt werden kann. Wenn die Input-Datei fehlt, wird eine Fehlermeldung angezeigt.

Wenn die Schaltfläche "Weiterleitung" deaktiviert ist, ist das Feld *XML-Input-Datei* der Zwischenkomponente aktiviert, wie oben gezeigt.

Beachten Sie den Unterschied zum Fall, in dem die Schaltfläche "Weiterleitung" der Komponente B aktiv ist. In diesem Fall wird das Feld "XML-Input-Datei" automatisch deaktiviert.

Vorschau:

Das Ergebnis des Mappings der Zwischenkomponente B auf die Zielkomponente C, d.h. die Reisekosten unter 1500.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <expense-report xsi:noNamespaceSchemaLocation="C:\test\mf-ExpRep-Target.xsd"
3   http://www.w3.org/2001/XMLSchema-instance">
4   <Person>
5     <First>Fred</First>
6     <Last>Landis</Last>
7     <Title>Project Manager</Title>
8     <Phone>123-456-78</Phone>
9     <Email>f.landis@nanonull.com</Email>
10  </Person>
11  <expense-item type="Travel" expto="Development">
12    <Date>2003-01-02</Date>
13    <Travel Trav-cost="337.88"/>
14    <description>Biz jet</description>
15  </expense-item>
16  <expense-item type="Travel" expto="Accounting">
17    <Date>2003-07-07</Date>
18    <Travel Trav-cost="1014.22"/>
19    <description>Ambassador class</description>
20  </expense-item>
21 </expense-report>
```

Anmerkung:

Wenn dieses Mapping über die Befehlszeile oder generierten Code ausgeführt wird, versucht MapForce, unabhängig davon, ob die Schaltfläche "Weiterleitung" in Komponente B aktiv ist oder nicht und unabhängig davon, welche Komponente für die Vorschau ausgewählt wurde, die Ausgabedaten von Komponente B und Komponente C zu generieren. Der Status der Vorschau-Schaltfläche hat keine Auswirkung.

Da die Einträge im Feld "XML-Input-Datei" und "XML-Output-Datei" (leer) nicht identisch sind, ist die Mapping-Kette unterbrochen und es kann keine Ausgabedatei für Komponente C generiert werden. Damit die Codegenerierung erfolgreich ist, müssen die Einträge in diesem beiden Feldern identisch sein.

Speichern des Mapping-Zwischenergebnisses

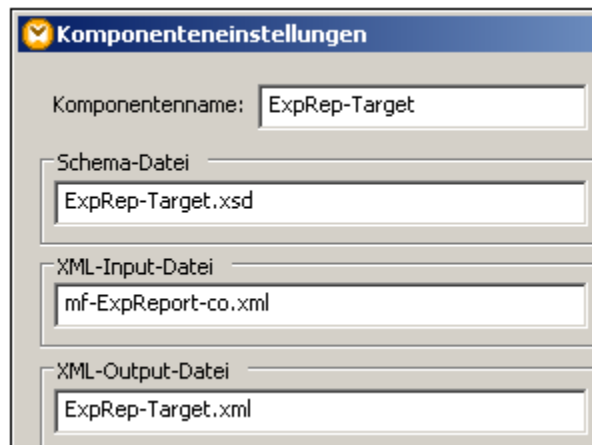
Um Zugriff auf die Zwischenkomponente zu erhalten, wenn die Schaltfläche "Weiterleitung" deaktiviert ist, muss das Ergebnis des Mappings von Komponente A auf B gespeichert werden. Der Name dieser Datei muss in das Feld *XML-Input-Datei* der Komponente B eingesetzt werden. Erst dann können die Daten in der Endkomponente C angezeigt werden.

So speichern Sie das Mapping-Zwischenergebnis in einer Datei:

1. Klicken Sie auf die Vorschau-Schaltfläche der Komponente B, um diese zu aktivieren und klicken Sie anschließend auf die Schaltfläche "Ausgabe".
2. Klicken Sie in der Symbolleiste der Ausgabevorschau auf die Schaltfläche "**Generierte**

Ausgabe speichern"  und geben Sie der XML-Datei einen Namen, z.B. mf-ExpReport-co.xml.

3. Doppelklicken Sie auf die Titelleiste der Komponente B, um das Dialogfeld **Komponenteneinstellungen** zu öffnen, kopieren Sie den Dateinamen in das Feld *XML-Input-Datei* und klicken Sie auf **OK**.



Bitte beachten Sie: Damit **Code generiert** werden kann und damit das Mapping über die Befehlszeile ausgeführt werden kann, **muss** sowohl der Name der Input- als auch der der Output-Datei identisch (und vorhanden) sein.

Anzeigen des Ergebnisses mit Hilfe von StyleVision

Wenn einer Zielkomponente eine SPS-Datei zugewiesen wurde, so werden die Ergebnisdaten bei Klick auf eines der Register HTML, RTF, PDF oder Word auf dem entsprechenden StyleVision-Register von MapForce angezeigt.



Personal Expense Report

Currency: ☒ Dollars ☐ Euros ☐ Yen **Currency** \$

☒ Detailed report

Employee Information

Fred	Landis	Project Manager
First Name	Last Name	Title
f.landis@nanonull.com		123-456-78
E-Mail		Phone

Expense List

Type	Expense To	Date (yyyy-mm-dd)	Expenses \$		Description
Travel	Development	2003-01-02	Travel 337.88	Lodging	Biz jet
Travel	Accounting	2003-07-07	Travel 1014.22	Lodging	Ambassador class

Mapping Database Query Output ☒ HTML ☒ RTF

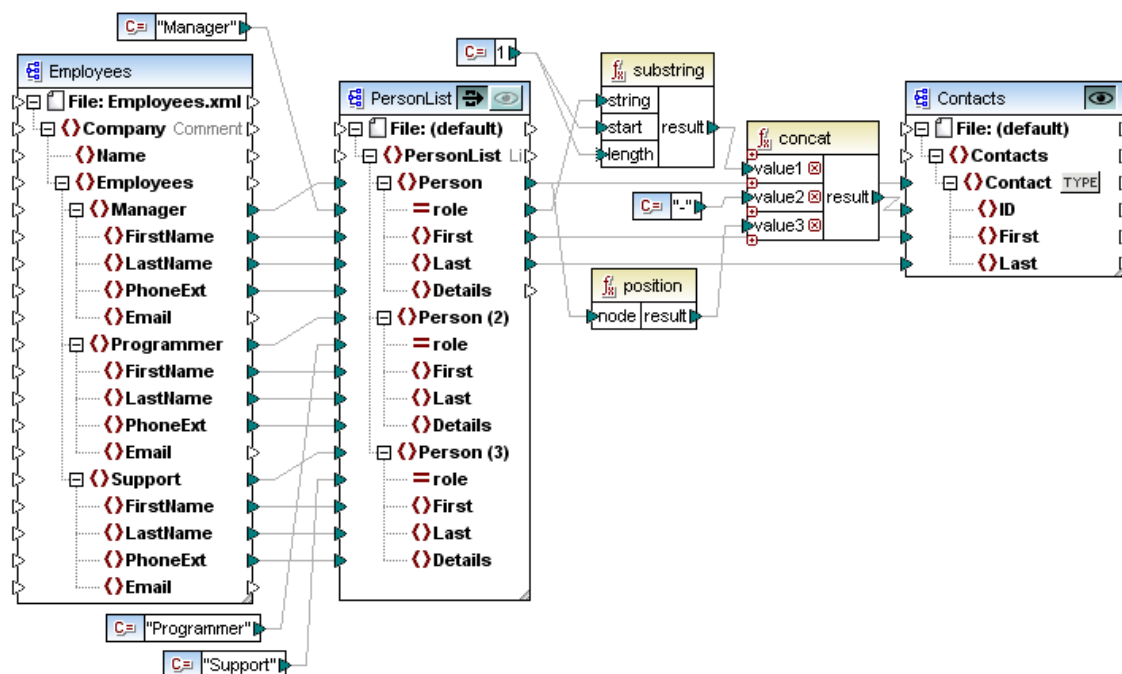
6.6.3 Beispiel für ein verkettetes Mapping

Sie finden das unten gezeigte Beispiel unter dem Namen **ChainedPersonList.mfd** im Ordner ...MapForceExamples.

Ziel:

Die Erstellung zweier Gruppen von Employee-Dokumenten (Angestellendokumenten), eine für die Personalabteilung und eine für die Buchhaltung.

- Im Dokument für die Buchhaltungsabteilung wird dem Angestellten eine eindeutige ID zugewiesen.
- Das Dokument für die Personalabteilung enthält die Personaldaten sowie die jeweilige Durchwahl (PhoneExt).



Komponenten:

Employees:

Die Instanzdatei employees.xml enthält vier Personen mit folgenden Berufstiteln (roles): manager, programmer und support.

PersonList: (die Ausgabe ist das Dokument für die Personalabteilung)

- Zu den Personendaten wird ein Attribut **role** hinzugefügt. Die Positionshierarchie aus der Komponente "Employees" wird entfernt.
- Die Schaltfläche "Weiterleitung" ist aktiv

Contacts: (ausgegeben wird das Dokument für die Buchhaltung)

- Zu den Contact-Daten wird ein Element "ID" hinzugefügt, um sicherzustellen, dass die Personendaten eindeutig sind.

Funktionsweise:

PersonList:

- Das Element "person" wird zweimal dupliziert, um die drei Arten von Berufsbezeichnungen, die es im Unternehmen gibt, zu ermöglichen.
- Die role -Namen werden mittels Konstantenkomponenten als Strings hinzugefügt, wobei die Konstantenkomponenten in derselben Reihenfolge verwendet werden wie in der Komponente "Employees".

Contacts:

- Die Funktion **substring** trennt den ersten Buchstaben des Attributs "role" ab und leitet ihn an die concat-Funktion weiter.
- Die Funktion **position** iteriert über alle Person-Nodes, weist ihnen eine fortlaufende Nummer zu (beginnend mit 1) und leitet sie an die concat-Funktion weiter.
- Die **concat** Funktion kombiniert das substring-Zeichen, einen Bindestrich (aus der Konstantenkomponente) und die Positionsnummer und leitet das Ergebnis an das Element "ID" der Contacts-Komponente weiter.

Ergebnis:

Die Komponente **PersonList** (Ausgabe: Dokument für die Personalabteilung)

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <PersonList xmlns:noNamespaceSchemaLocation="C:/DOCUMENTS AND SETTINGS/ALTOVA/ALTOVA MAPFORCE/ALTOVA MAPFORCE.XML"
3    http://www.w3.org/2001/XMLSchema-instance">
4    <Person role="Manager">
5      <First>Vernon</First>
6      <Last>Callaby</Last>
7      <Details>582</Details>
8    </Person>
9    <Person role="Programmer">
10     <First>Frank</First>
11     <Last>Further</Last>
12     <Details>471</Details>
13   </Person>
14   <Person role="Support">
15     <First>Loby</First>
16     <Last>Matise</Last>
17     <Details>963</Details>
18   </Person>
19   <Person role="Support">
20     <First>Susi</First>
21     <Last>Sanna</Last>
22     <Details>753</Details>
23   </Person>
24 </PersonList>

```

Die Komponente **Contacts** (Ausgabe: Dokument für die Buchhaltung)

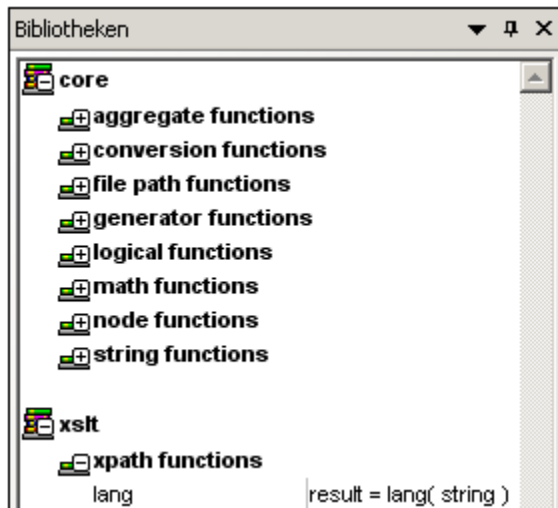
```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Contacts xmlns:noNamespaceSchemaLocation="C:/DOCUMENTS AND SETTINGS/ALTOVA/ALTOVA MAPFORCE/ALTOVA MAPFORCE.XML"
3    http://www.w3.org/2001/XMLSchema-instance">
4    <Contact>
5      <ID>M-1</ID>
6      <First>Vernon</First>
7      <Last>Callaby</Last>
8    </Contact>
9    <Contact>
10     <ID>P-2</ID>
11     <First>Frank</First>
12     <Last>Further</Last>
13   </Contact>
14   <Contact>
15     <ID>S-3</ID>
16     <First>Loby</First>
17     <Last>Matise</Last>
18   </Contact>
19   <Contact>
20     <ID>S-4</ID>
21     <First>Susi</First>
22     <Last>Sanna</Last>
23   </Contact>
24 </Contacts>

```

6.7 Arbeiten mit Funktionen

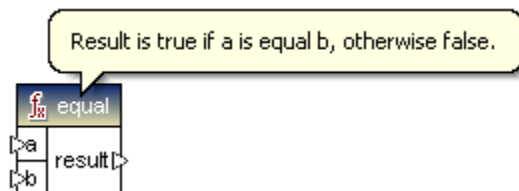
Die in der ausgewählten Sprache (XSLT/XSLT2, BUILT-IN XQ, Java, C# oder C++) verfügbaren Funktionen werden im Fenster "Bibliotheken" angezeigt. Mit Hilfe der Ein- und Ausklappsymbole können Sie die Funktionen der einzelnen Bibliotheken ein- und ausblenden.



XSLT ausgewählt

Funktions-Tooltips:

Erklärender Text zu den einzelnen Funktionen, der im Bibliotheksfenster angezeigt wird, kann nun durch Klicken auf das Symbol "Tipps anzeigen"  in der Symbolleiste ein- oder ausgeblendet werden. Wenn Sie den Mauszeiger über eine Funktionsbezeichnung platzieren, werden Informationen zu dieser Funktion angezeigt.



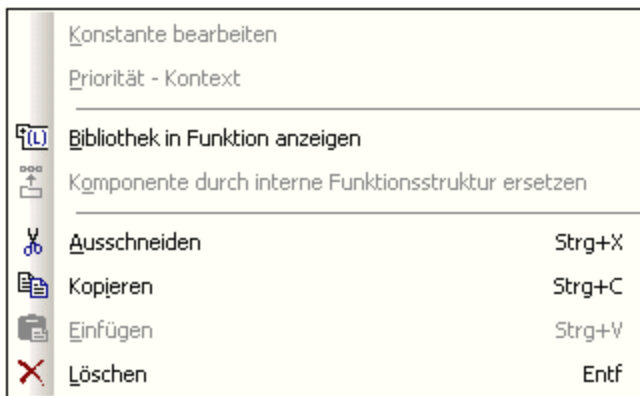
So verwenden Sie eine Funktion im Mapping-Fenster:

1. Wählen Sie zuerst die **Programmiersprache** aus, für die Code erzeugt werden soll, indem Sie in der Symbolleiste auf eines der Ausgabecode-Symbole klicken.
2. Klicken Sie auf den **Funktionsnamen** und ziehen Sie ihn in das Mapping-Fenster.
3. Verbinden Sie die Eingabe- und Ausgabeparameter der verschiedenen Symbole durch Drag & Drop miteinander.

Hinweis: Wenn Sie den Mauszeiger im Bibliotheksfenster über dem Ausdruck "result = xxx" positionieren, wird ein Tooltip mit einer näheren Beschreibung der Funktion angezeigt.

Funktions-Kontextmenü:

Durch einen Rechtsklick auf eine Funktion im Mapping-Fenster wird das Kontextmenü geöffnet.



Prioritätskontext

Bei Anwendung einer Funktion auf verschiedene Datenelemente eines Schemas oder einer Datenbank muss MapForce den Kontext-Node kennen. Alle anderen Elemente werden dann relativ zu diesem verarbeitet. Zu diesem Zweck wird das Datenelement (oder der Node) als Prioritätskontext definiert. Das jeweilige Symbol wird durch einen Kreis markiert. Ein Beispiel dazu finden Sie im Abschnitt "Referenz" unter "[Prioritätskontext](#)".



Bibliothek in Funktionstitelleiste anzeigen

Zeigt den Namen der Bibliothek im Titel der Funktionskomponente an.

Komponente durch interne Funktionsstruktur ersetzen

Ersetzt die benutzerdefinierte Komponente/Funktion durch ihre Bestandteile. Steht für Funktionen nicht zur Verfügung.

Ausschneiden/Kopieren/Einfügen/Löschen

Mit Hilfe der MS Windows-Standardbearbeitungsbefehle können Sie alle im Mapping-Fenster angezeigten Komponenten oder Funktionen ausschneiden, kopieren, usw. Alle Konnektoren mit Ausnahme derer, die ersetzt werden müssten, bleiben erhalten.

Eigenschaften

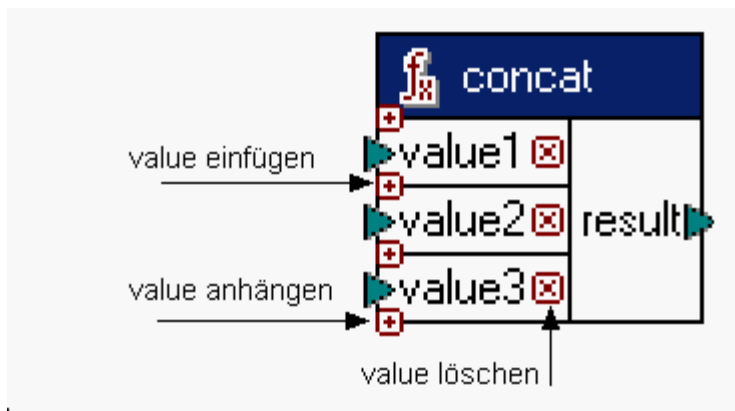
Steht für Funktionen nicht zur Verfügung.

Erweiterbare Funktionen

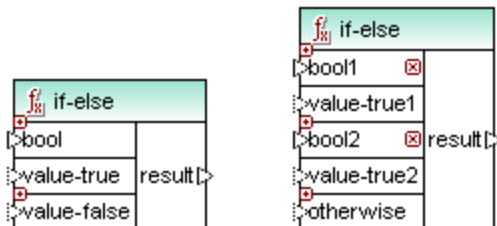
Eine Reihe von Funktionen aus den Funktionsbibliotheken kann erweitert werden, z.B: die Funktionen concat, "logical-and", "logical-or" und IF-ELSE. Die Parameter dieser Arten von Funktionen können je nach Belieben eingefügt/angehängt und gelöscht werden.

Wenn Sie auf das Plus-Symbol klicken, wird dieselbe Parameterart eingefügt oder angehängt. Wenn Sie auf das Häkchen klicken, wird der Parameter gelöscht.

Beachten Sie bitte. Wenn Sie einen Konnektor auf das Plus-Symbol ziehen, wird der Parameter automatisch eingefügt/angehängt und verbunden.



Die IF-Testparameter der [IF-Else](#) Funktion können auf dieselbe Art erweitert werden.



6.8 Vorschau auf das Transformationsergebnis

Mit Hilfe des Built-In Ausführungsprozessors können Sie das Ergebnis einer Transformation sofort in der Vorschau anzeigen und speichern, ohne dass vorher Programmcode generiert, kompiliert und anschließend die Ergebnisse angezeigt werden müssen.

Dazu müssen Sie einfach - unabhängig davon, welche Ausgabesprache Sie gewählt haben, auf die Schaltfläche **Ausgabe** klicken. Das Fenster **Ausgabe** unterstützt auch den Befehl "Suchen", um nach allen benötigten XML-Daten oder SQL-Anweisungen suchen zu können.

Wenn Sie auf die Schaltfläche **BUILTIN**  klicken, wird der Built-In-Ausführungsprozessor explizit als Ausgabeziel ausgewählt. Mit Hilfe des Built-In-Ausführungsprozessors wird auch die Vorschau des Ergebnisses angezeigt, wenn eine der Programmiersprachen (Java, C#, C++) ausgewählt ist. Nachdem Sie auf die Schaltfläche BUILTIN und anschließend auf **Ausgabe** geklickt haben, stehen alle Standardoptionen des Ausgabefenster zur Verfügung: also **Ausgabedatei validieren**, **Ausgabedatei speichern...**, **Alle Ausgabedateien speichern...**, usw.

MapForce lässt sich auch über die [Befehlszeile](#) starten und kann mit Hilfe des Parameters /BUILTIN auch ohne die Erzeugung von Zwischencode ein Ergebnis produzieren.

Streaming:

Der Built-In-Ausführungsprozessor ist das einzige Ziel, das XML, CSV und FLF **Streaming** unterstützt. Dies bedeutet, dass es für die **Ausgabedateien** keine Größenbeschränkung gibt. Bei sehr großen Ausgabedateien wird am unteren Rand des Ausgabefensters eine Schaltfläche "Mehr laden.." angezeigt. Wenn Sie darauf klicken, wird ein großer Abschnitt an die bereits sichtbaren Daten angehängt. Die Schaltfläche "Pretty-Print" wird aktiv, sobald die komplette Datei in das Ausgabefenster geladen wurde.

Digitale Signaturen:

MapForce unterstützt die Erstellung von digitalen XML-Signaturen für XML- und XBRL-Ausgabedateien. Digitale Signaturen können nur erstellt werden, wenn das BUILTIN das Ausgabeziel ist. Eine Signatur wird erstellt, wenn Sie auf die Schaltfläche "Ausgabe" klicken und die Ergebnisdatei gespeichert wird. Nähere Informationen dazu finden Sie unter [Digitale Signaturen](#).

Vorschau auf die Transformationsausgabe

Im Menü **Ausgabe** oder bei Auswahl einer der Schaltflächen in der Symbolleiste "Auswahl der Sprache" (*Abbildung unten*) können Sie definieren, welche Sprach MapForce zum Transformieren der Daten der Input-Datei verwenden soll.



Die folgenden Optionen stehen zur Verfügung:

- XSLT
- XSLT2

Auf dem Register "XSLT/XSLT2" sehen Sie eine Vorschau auf den XSLT-Code. Er wird mit der Menüoption **Datei | Code generieren in | XSLT 1.0** bzw. **XSLT 2.0** generiert und gespeichert.

- XQuery

Bitte beachten Sie, dass die Ausführungsgeschwindigkeit von generiertem XQuery 1.0-Code deutlich schneller ist als die von generiertem XSLT 1.0 / 2.0-Code.

- Java
- C#
- C++

Generierter Programmcode wie z.B. Java, C# oder C++ ist natürlich noch schneller als XQuery, da er vor der Ausführung kompiliert wird.

Anzeigen einer Vorschau auf den Programmcode

Bevor der Programmcode generiert wird, empfiehlt es sich, das Ergebnis von XSLT/ XQuery mit Hilfe des Built-In-Ausführungsprozessor anzuzeigen.

So zeigen Sie eine Vorschau eines XSLT- / XQuery-Ergebnisses an:

Öffnen Sie zuerst die Datei *.mfd in MapForce:

1. Wählen Sie eine der folgenden Methoden:
 - Klicken Sie auf die Schaltfläche **XSLT**, um eine Vorschau des generierten XSLT 1.0-Codes zu sehen.
 - Klicken Sie auf die Schaltfläche **XSLT2**, um eine Vorschau des generierten XSLT 2.0-Codes zu sehen.
 - Klicken Sie auf die Schaltfläche **XQuery**, um eine Vorschau des generierten XQuery-Codes zu sehen.
2. Klicken Sie auf die Schaltfläche **Ausgabe**, um eine Vorschau des Mapping-Ergebnisses anzuzeigen.

Generieren von XSLT- und XQuery-Code

XSLT 1.0- und XSLT 2.0-Programmcode kann von MapForce mit Hilfe des AltovaXML-Prozessors, der im AltovaXML Paket enthalten ist, generiert werden. Der Prozessor kann auch direkt in XMLSpy aufgerufen und die Codegenerierung kann über den Menübefehl **XSL/XQuery | XSL-Transformation** durchgeführt werden.

MapForce generiert XQuery 1.0-Programmcode mit Hilfe des AltovaXML-Prozessors, der im AltovaXML Paket enthalten ist. Der Prozessor kann auch direkt in XMLSpy aufgerufen und die Codegenerierung kann über den Menübefehl **XSL/XQuery | XQuery-Ausführung** durchgeführt werden. Um XQuery-Code in MapForce zu generieren und zu speichern, verwenden Sie den Menübefehl **Datei | Code generieren in | XQuery**.

So laden Sie das AltovaXML-Paket herunter:

1. Gehen Sie zu http://www.altova.com/download_components.html.
2. Klicken Sie auf den Link "Informieren Sie sich über AltovaXML und laden Sie das Tool jetzt herunter!"
3. Wählen Sie die gewünschte AltovaXML-Edition aus und installieren Sie sie.

So generieren Sie XSLT-Code:

1. Wählen Sie die Menüoption **Datei | Code generieren in | XSLT 1.0 (oder XSLT 2.0)**.
3. Wählen Sie den Ordner, in dem die generierte XSLT-Datei abgelegt werden soll. Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
4. Navigieren Sie zum angegebenen Ordner, wo Sie die generierte XSLT-Datei finden.

So generieren Sie XQuery-Code:

1. Wählen Sie die Menüoption **Datei | Code generieren in | XQuery**.
3. Wählen Sie den Ordner, in dem die generierte XQuery-Datei abgelegt werden soll und klicken Sie auf OK. Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
4. Navigieren Sie zum angegebenen Ordner, wo Sie die generierte XQuery-Datei finden.

So generieren Sie Programmcode:

1. Wählen Sie die Menüoption **Datei | Code generieren in | Java, C# (Sharp), C++**.
3. Wählen Sie den Ordner, in dem die Programmdateien gespeichert werden sollen.
Es wird eine Meldung angezeigt, dass die Generierung erfolgreich war.
4. Kompilieren Sie den Code mit Ihrem Compiler und führen Sie ihn aus.

Bitte beachten Sie: Projektdateien für verschiedene IDEs und Build-Systeme werden von MapForce automatisch generiert. Nähere Informationen dazu finden Sie im Abschnitt [JDBC-Treiberinstallation](#) und im Abschnitt zum Code Generator.

Transformieren der XML-Quelldatei mit Hilfe des generierten XSLT- oder XQuery-Codes.

Wenn Sie XSLT- oder XQuery-Code generieren, wird eine Batch-Datei namens DoTransform.bat, die die XML-Datei mit Hilfe von AltovaXML transformiert, generiert. Diese Datei wird in dem Ordner gespeichert, in dem Sie auch die XSLT- bzw. XQuery-Datei speichern.

So transformieren Sie die XML-Datei mittels der generierten XSLT-/XQuery-Datei:

1. Laden Sie den kostenlosen AltovaXML-Prozessor von der [AltovaXML Download-Seite](#) herunter und installieren Sie ihn. AltovaXML wird standardmäßig im Verzeichnis c:\Program Files\Altova\AltovaXML2012\ installiert.
2. Starten Sie die Batch-Datei **DoTransform.bat**, die sich im zuvor angegebenen Ausgabeordner befindet.
Daraufhin wird im aktuellen Ordner die Ausgabedatei **ExpReport-Target.xml** generiert.

Anmerkung: Eventuell müssen Sie den AltovaXML Installationspfad zur Pfadvariablen der Umgebungsvariablen hinzufügen.

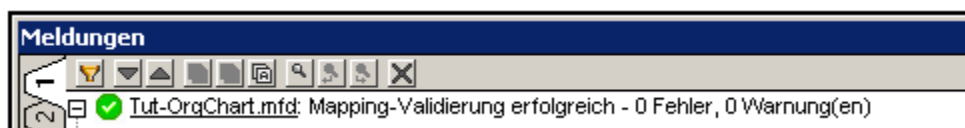
6.9 Validieren von Mappings und Mapping-Ausgabe

Es müssen nicht alle Funktionen oder Komponenten gemappt werden. Das Mapping-Register ist ein Arbeitsbereich, in den beliebige verfügbare Komponenten platziert werden können. XSLT 1.0, XSLT 2-, XQuery-, Java-, C#- oder C++-Code wird nur für die Komponenten generiert, für die gültige Verbindungen existieren.

Bei **frei stehenden** Komponenten werden keinerlei Fehlermeldungen oder Warnungen erzeugt.

Bei **teilweise verbundenen** Komponenten können zwei Arten von Warnungen generiert werden:

- Wenn das **Input-Symbol** einer Funktionskomponente nicht verbunden ist, wird eine Fehlermeldung generiert und die Transformation angehalten.
- Wenn das **Output-Symbol** der Funktion nicht verbunden ist, wird eine Warnung generiert und der Transformationsvorgang fortgesetzt. Die störende Komponente und ihre Daten werden ignoriert und nicht auf das Zieldokument gemappt.



Sie können mehrere Meldungsregister verwenden, wenn Ihr Projekt mehrere separate Mapping-Dateien enthält. Klicken Sie auf eines der nummerierten Register im Fenster "Meldungen" und klicken Sie auf das Register "Vorschau" um ein anderes Mapping in Ihrem Projekt zu sehen. Die Validierungsmeldung wird nun auf dem ausgewählten Register angezeigt. Die ursprüngliche Meldung bleibt jedoch auf Register 1 erhalten.

Verwenden Sie die verschiedenen Symbole auf dem Register "Meldungen" zu folgenden Zwecken:

- Filtern der Meldungsarten, Fehler oder Warnmeldungen
- Scrollen durch die Einträge
- Kopieren des Meldungstexts in die Zwischenablage
- Suchen eines bestimmten Strings in einer Meldung
- Löschen der Meldungen im Fenster "Meldungen"

So validieren Sie ein Mapping:

- Klicken Sie in der Symbolleiste der Applikation auf das Symbol "Mapping validieren"



oder wählen Sie den Menübefehl **Datei | Mapping validieren**.

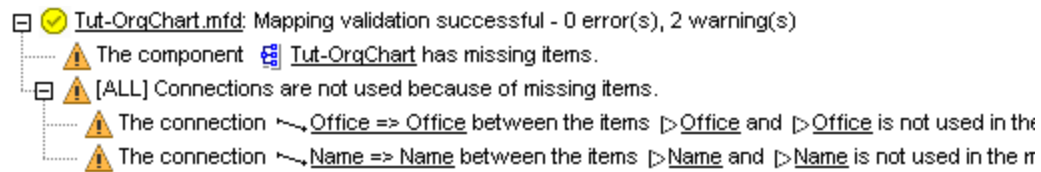
Im Fenster "Meldungen" wird eine Validierungsmeldung angezeigt.

Validierungsmeldungen:

Validierungsmeldungen werden im Fenster "Validierungen" angezeigt und zeigen an, ob die Validierung erfolgreich war oder nicht. Zusätzlich dazu werden Fehlermeldungen, Warnungen und Informationen über das Mapping angezeigt. Es gibt zwei Arten von Validierungsmeldungen:

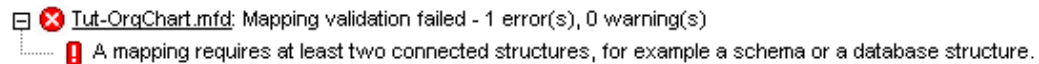
- Validierung erfolgreich - X Fehler, Y Warnung(en).
 Tut-OrgChart.mfd: Mapping validation successful - 0 error(s), 0 warning(s)
 Information: The output component  Tut-Person has no output file name set. A default file name

Warnungen weisen auf ein Problem hin. Das Mapping und die Transformationsvorschau werden dadurch aber nicht abgebrochen. Ein Mapping kann daher 0 Fehler und n Warnungen aufweisen.



- Validierung fehlgeschlagen - X Fehler, Y Warnung(en).

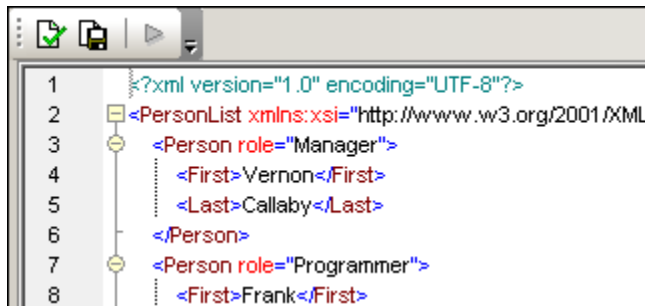
Fehler stoppen die Transformation und erzeugen eine Fehlermeldung. Bei Fehlern dieser Art ist eine XSLT, XQuery- oder Ausgabe nicht möglich. Wenn Sie im Fenster "Meldungen" auf eine Validierungsmeldung klicken, wird das Symbol der Komponente, bei der das Problem aufgetreten ist, im Mapping-Fenster markiert.



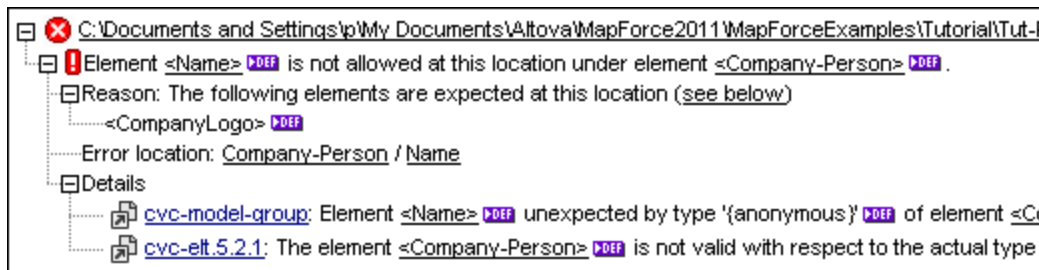
Validieren der Mapping-Ausgabe

Wenn Sie auf die Schaltfläche "Ausgabe" klicken, werden die Daten mit Hilfe des MapForce, XSLT 1.0/XSLT 2.0- oder XQuery-Prozessors transformiert und es wird ein Ergebnis in der Textansicht erzeugt.

Werden die Daten auf ein XML-Schema gemappt, kann das erzeugte XML-Dokument gegen das zugrunde liegende Schema validiert werden. Wenn es sich bei der Zielkomponente um eine EDI-Datei handelt, wird die Ausgabe gegen die EDI-Spezifikation validiert. Nähere Informationen dazu siehe: [UN/EDIFACT und ANSI X12 als Zielkomponenten](#).



Das Ergebnis der Validierung wird im Fenster "Meldungen" angezeigt. Wenn die Validierung nicht erfolgreich war, so enthält dieses Fenster detaillierte Informationen über die aufgetretenen Fehler (siehe Abbildung unten).



Die Validierungsmeldung enthält eine Reihe von Hyperlinks. Wenn Sie darauf klicken erhalten Sie weitere ausführliche Informationen::

- Wenn Sie auf den Dateipfad klicken, wird die Ausgabe der Transformation im Ausgabefenster von MapForce geöffnet.

- Wenn Sie auf den Link <ElementName> klicken, wird das Element im Ausgabefenster markiert.
- Wenn Sie auf das Symbol  klicken, wird die Definition des Elements in [XMLSpy](#) geöffnet (falls das Programm installiert ist).
- Wenn Sie auf die Hyperlinks im Unterabschnitt "Details" klicken (z.B., cvc-model-group) wird eine Beschreibung der entsprechenden Validierungsregel auf der <http://www.w3.org/> Website geöffnet.

So validieren Sie die Mapping-Ausgabe:

- Klicken Sie auf die Schaltfläche "Validieren" , um das Dokument gegen das Schema zu validieren. Daraufhin wird entweder die Meldung "Das ausgegebene XML-Dokument ist gültig" oder eine Meldung mit der Erklärung des Fehlers angezeigt.

 ...\\Tutorial\\ExpReport-Target.xml: Output file validation successful. - 0 error(s), 0 warning(s)

Bitte beachten Sie:

Über das Feld **Schema /DTD-Referenz hinzufügen** im Dialogfeld "Komponenteneinstellungen" können Sie den Pfad der referenzierten XML-Schemadatei zum Root-Element der XML-Ausgabedatei hinzufügen.

6.10 Schleifen, Gruppen und Hierarchien

Es gibt verschiedene Möglichkeiten, Quell- und Zielhierarchien in Schleifen zu verarbeiten. Sie können definieren, wie bestimmte Gruppen von Daten in Form von Schleifen verarbeitet oder gruppiert werden sollen.

Unter den folgenden Links finden Sie nähere Informationen zu diesen Themen. Beachten Sie bitte, dass diese Beispiele nicht nacheinander aufgelistet sind, sondern in verschiedenen Kapiteln dieser Dokumentation behandelt werden.

[Mappen von XML-Daten auf CSV-Dateien oder Textdateien mit fester Länge](#)

[Mappen von CSV-Dateien auf XML-Dateien](#)

[Erstellen von Hierarchien anhand von CSV-Dateien und Textdateien mit fester Länge](#)

[Lookup-Tabelle für die Wertezuordnung](#)

[Mappen mehrerer Tabellen auf eine XML-Datei](#)

[Erstellen von Datenbankbeziehungen mit Hilfe von MapForce](#)

[Prioritätskontext](#)

6.11 Mapping-Regeln und -Strategien

MapForce mappt Daten im Allgemeinen auf intuitive Art, doch gibt es einige Szenarien, in denen die erzeugte Ausgabe scheinbar zu viele oder zu wenige Datenelemente hat. Diese Fälle werden in diesem Abschnitt behandelt.

Allgemeinregel

Im Allgemeinen bedeutet jede Verbindung zwischen einem Quell- und einem Zieldatenelement: Erstelle für jedes Quelldatenelement ein Zieldatenelement.

Wenn der Quell-Node simple Content (z.B. String, Ganzzahl usw.) enthält und im Ziel-Node simple Content gestattet ist, so wird der Inhalt kopiert und der Datentyp wird gegebenenfalls konvertiert.

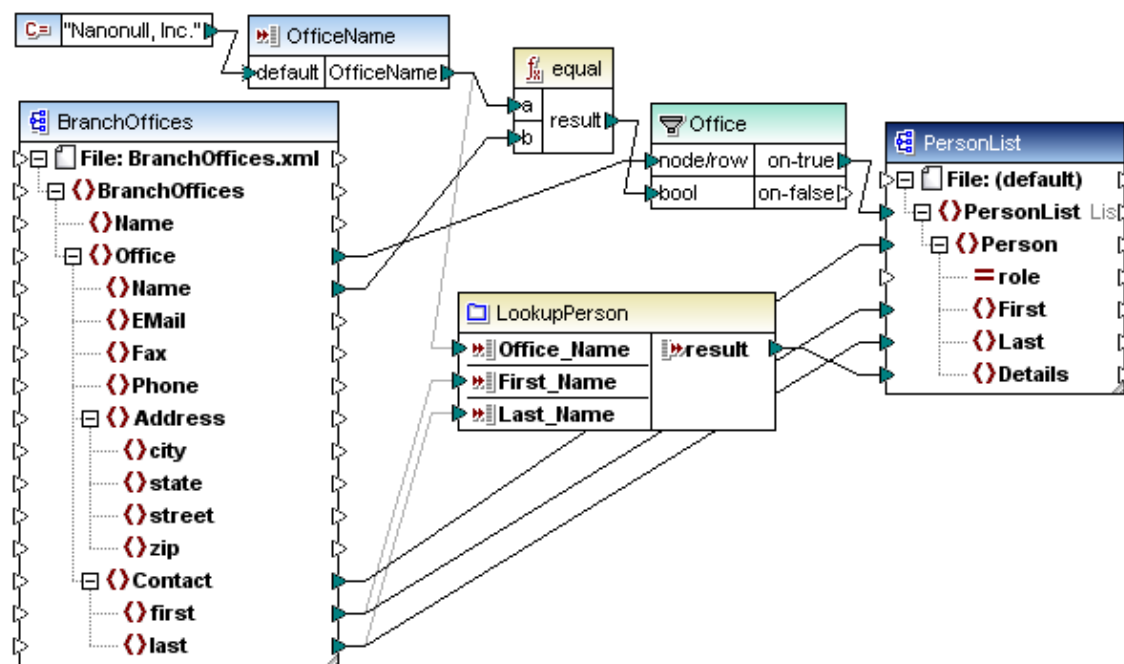
Für diese Regel gibt es einige Ausnahmen, jedoch gilt sie im Allgemeinen für alle Verbindungen.

Die Kontext-Datenelemente und aktuelle Datenelemente

MapForce zeigt die Struktur eines Schemas, einer Datenbank oder EDI-Datei in Form einer Hierarchie mapbarer Datenelemente in der Komponente an. Jeder dieser Nodes kann in der Instanzdatei bzw. der Datenbank viele (oder keine) Instanzen haben.

Beispiel: Wenn Sie sich in der Datei PersonListByBranchOffice.mfd die Quellkomponente ansehen, so gibt es dort nur einen einzigen Node namens **first** (unter **Contact**). In der Instanzdatei **BranchOffices.xml** gibt es unterhalb unterschiedlicher übergeordneter **Office**-Nodes mehrere **first**-Nodes und **Contact**-Nodes mit jeweils unterschiedlichem Inhalt.

Es hängt vom aktuellen **Kontext** (des **Ziel**-Node) ab, welche Quell-Nodes tatsächlich ausgewählt werden, damit ihre Daten über den Konnektor in die Zielkomponente/das Zieldatenelement kopiert werden.



Dieser Kontext definiert sich durch den **aktuellen Ziel-Node** und seine Verbindungen mit

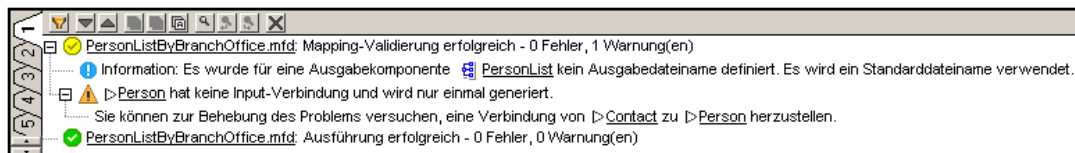
übergeordneten Nodes.

- Am Anfang enthält der Kontext nur die Quellkomponenten, aber keine spezifischen Nodes. MapForce verarbeitet beim Mapping zuerst den **Ziel-Node** (PersonList) und arbeitet sich anschließend durch die Hierarchie nach unten.
- Der Konnektor zum **Ziel-Node** wird zurück zu allen direkt oder indirekt (über Funktionen zwischen den zwei Komponenten) damit verbundenen Quell-Nodes verfolgt. Die Quell-Nodes und die Ergebnisse der Funktionen werden zum Kontext für diesen Node hinzugefügt.
- Für jeden neuen Ziel-Node wird ein neuer Kontext angelegt, der anfangs alle aktuellen Datenelemente des Kontexts des übergeordneten Nodes enthält. Gleichrangige Ziel-Nodes sind daher voneinander unabhängig, haben aber Zugriff auf alle Quelldaten der übergeordneten Datenelemente.

Angewendet auf das obige Beispielmapping (**PersonListByBranchOffice.mfd**):

- Die Verbindung von **Office** durch den Filter (Office) zu **PersonList** definiert ein **einziges** Büro (office) als den Kontext für das gesamte Zieldokument (weil PersonList des Root-Element der Zielkomponente ist). Der Büroname wird von der Input-Komponente geliefert, deren Standardinhalt "Nanonull, Inc." ist.
- Die Filterbedingung wirkt sich auf alle Verbindungen zu den **Nachfahren** bzw. Daten des Root-Elements "PersonList" aus, da sich das ausgewählte Büro im Kontext befindet.
- Durch die Verbindung von **Contact** zu **Person** wird eine Ziel-Person **pro** Contact-Element der XML-Quelldatei (allgemeine Regel) angelegt. Für jede Person wird ein bestimmter Contact zum Kontext, anhand dessen die untergeordneten Nodes von Person erstellt werden, hinzugefügt.
- Der Konnektor von **first** zu **First** wählt den Vornamen des aktuellen Kontakts (Contact) aus und schreibt ihn in das Zieldatenelement "First".

Wenn wir den Konnektor von **Contact** zu **Person** weglassen, würde nur ein Person-Element mit mehreren First-, Last- und Detail-Nodes erstellt werden, was nicht das gewünschte Ergebnis erzielt. In solchen Fällen gibt MapForce eine Warnmeldung aus und schlägt eine Lösung für das Problem vor: "Sie können zur Behebung des Problems versuchen, eine Verbindung von Contact zu Person herzustellen".



Sequenzen

MapForce zeigt die Struktur eines Schemas, einer Datenbank oder EDI-Datei in Form einer Hierarchie mapbarer Datenelemente in der Komponente an.

Je nach dem (Ziel)-Kontext kann ein **mapbares Datenelement** einer **Quellkomponente** Folgendes repräsentieren:

- einen **einzelnen Instanz-Node** der zugewiesenen Input-Datei (oder Datenbank)
- eine **Sequenz** von 0 bis zu mehreren **Instanz-Nodes** der Input-Datei (oder Datenbank)

Wenn eine Sequenz mit einem Ziel-Node verbunden wird, wird eine Schleife erstellt, mit der so viele Ziel-Nodes angelegt werden, wie Quell-Nodes vorhanden sind.

Wenn ein **Filter** zwischen die Sequenz und den Ziel-Node platziert wird, wird die Boolesche Bedingung für jeden Input-Node, also für jedes Datenelement in der Sequenz, überprüft. Genauer gesagt, wird überprüft, ob es mindestens eine Boolesche Bedingung in jeder Sequenz gibt, deren Ergebnis "true" ist. Die Prioritätskontext-Einstellung kann sich auf die Reihenfolge der Auswertung auswirken. Siehe unten.

Wie bereits oben erwähnt, werden Filterbedingungen automatisch auf alle untergeordneten Nodes angewendet.

Hinweis: Wenn im Quellschema angegeben ist, dass ein bestimmter Node nur genau einmal vorkommt, kann MapForce die Schleife entfernen und nur das erste Datenelement verwenden, weil ja bekannt ist, dass der Node vorhanden sein muss. Diese Optimierung kann im Dialogfeld "Komponenteneinstellungen" deaktiviert werden (Kontrollkästchen "Input-Verarbeitungsoptimierungen auf Basis von min/maxOccurs aktivieren").

Funktions-Inputs (von normalen Funktionen, die keine Sequenz-Funktionen sind) funktionieren ähnlich wie Ziel-Nodes: Wenn eine Sequenz mit einem solchen Input verbunden wird, wird eine Schleife rund um den Funktionsaufruf erstellt, damit **so viele Ergebnisse** erzeugt werden, wie es Datenelemente in der Sequenz gibt.

Wenn eine Sequenz mit mehr als einem solchen Funktions-Input verbunden wird, erzeugt MapForce verschachtelte Schleifen, die das **kartesische Produkt** aller Inputs verarbeiten. Gewöhnlich ist dies nicht erwünscht. Daher sollte nur eine einzige Sequenz mit mehreren Datenelementen mit einer Funktion (und allen anderen Parametern, die an einzelne aktuelle Datenelemente aus übergeordneten Nodes oder anderen Komponenten gebunden sind) verbunden werden.

Hinweis: Wenn eine leere Sequenz mit einer solchen Funktion (z.B. concat) verbunden ist, erhalten Sie als Ergebnis eine **leere Sequenz** und somit keinen einzigen Output Node. Wenn Ihre Ausgabedatei kein Ergebnis enthält, weil es keine Input-Daten gibt, können Sie mit Hilfe der "substitute-missing" Funktion einen Ersatzwert einfügen.

Funktionen mit Sequenz-Inputs sind die einzigen Funktionen, die ein Ergebnis erzeugen können, wenn die Input-Sequenz **leer** ist:

- **"exists"**, "not-exists" und "substitute-missing" (sowie "is-not-null", "is-null" und "substitute-null", die Aliasnamen für die erstgenannten drei Funktionen sind)
- **aggregate** Funktionen ("sum", "count" usw.)
- **reguläre** benutzerdefinierte Funktionen, die Sequenzen unterstützen (also nicht-inline-Funktionen),

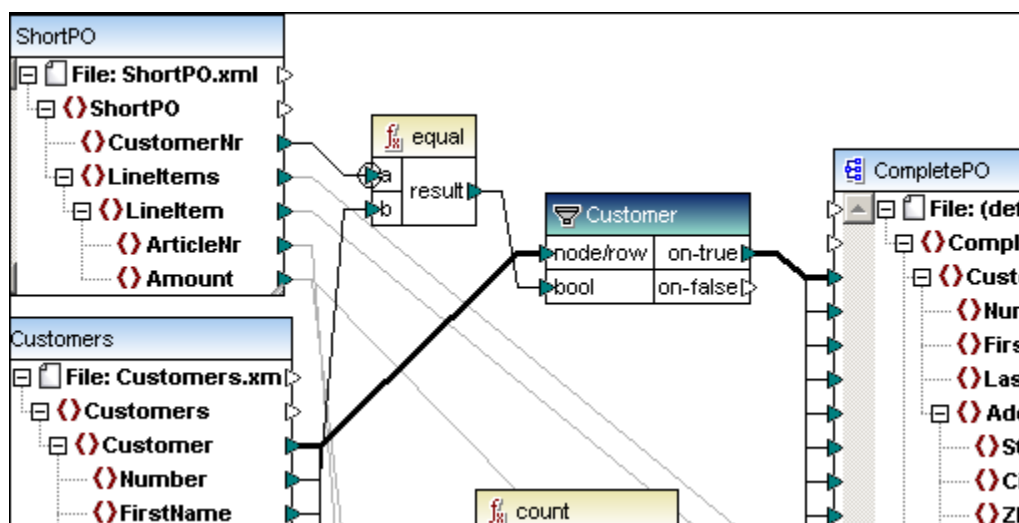
Der Sequenz-Input für derartige Funktionen wird immer unabhängig vom aktuellen Ziel-Node im Kontext seiner übergeordneten Nodes ausgewertet. Dies bedeutet auch, dass alle Filter- oder

SQL-Where-Komponenten, die mit solchen Funktionen verbunden sind, sich nicht auf andere Verbindungen auswirken.

Prioritätskontext

Im Allgemeinen werden Funktionsparameter von oben nach unten ausgewertet, es kann aber mit Hilfe der Einstellung "**Prioritätskontext**" auch ein Parameter definiert werden, der vor allen anderen ausgewertet werden soll.

In Funktionen, die mit dem Booleschen Input von **Filterbedingungen** verbunden sind, wirkt sich der Prioritätskontext nicht nur auf die Vergleichsfunktion selbst, sondern auch auf die Auswertung des Filters aus, daher können auch zwei Quellsequenzen (siehe CompletePO.mfd, CustomerNo und Number) miteinander verbunden werden.

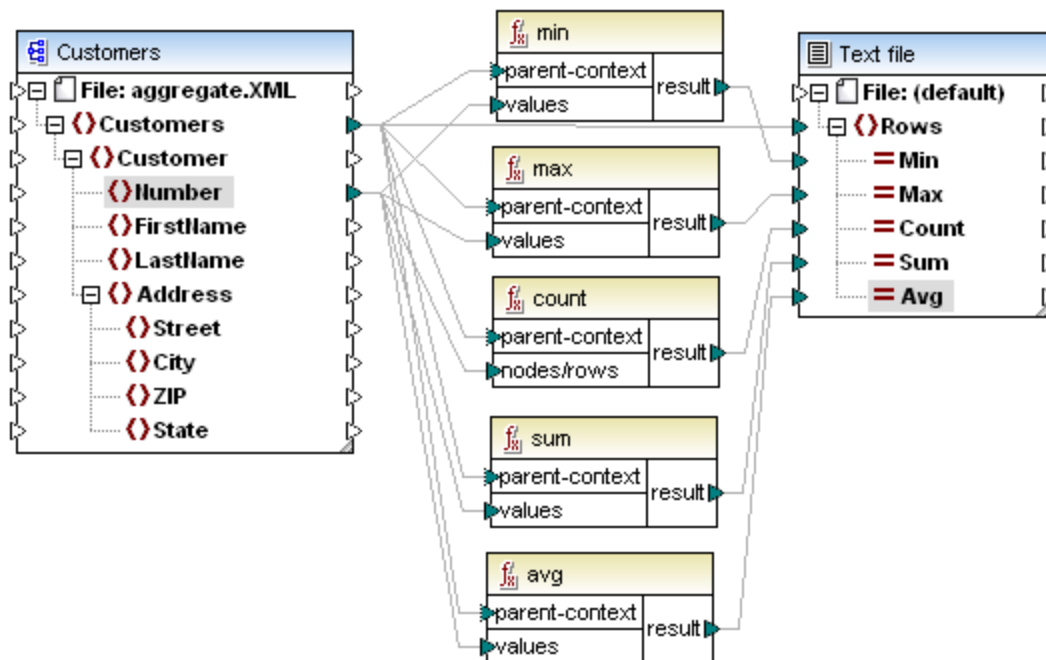


In diesem Beispiel wird ShortPO/CustomerNr aufgrund des Prioritätskontexts ausgewertet, bevor durch die Customer Nodes aus der Komponente "Customers" iteriert wird und diese gefiltert werden. Siehe Beispiel [Prioritätskontext Node/Datenelement](#).

Außerkräftsetzung des Kontexts

Einige [aggregate-Funktionen](#) haben einen optionalen "parent-context" Input.

Wenn dieser Input nicht verbunden wird, hat dies keine Auswirkung und die Funktion wird innerhalb des normalen Kontexts für Sequenz-Inputs (also im Kontext des übergeordneten Node des Ziel-Node) ausgewertet.



Wenn der parent-context-Context-Input mit einem Quell-Node verbunden ist, wird die Funktion **für jeden** "parent-context" Node ausgewertet und für jede Instanz wird ein separates Ergebnis erzeugt.

Ausnahmen für die allgemeine Regel (Erstelle für jedes Quelldatenelement ein Zieldatenelement)

- Ein [XML-Root-Zielelement](#) wird immer nur ein einziges Mal erstellt. Wenn eine Sequenz damit verbunden wird, so wird nur der Inhalt des Elements wiederholt, nicht aber das Root-Element selbst. Das Ergebnis ist aber unter Umständen nicht immer dem Schema gemäß gültig. Wenn auch Attribute des Root-Elements verbunden werden, so schlägt die XML-Serialisierung zur Laufzeit fehl. Daher sollten Sie Sequenzen nicht mit einem Root-Element verbinden. Um mehrere Ausgabedateien zu erzeugen, verbinden Sie die Sequenz statt dessen über eine Funktion, die Dateinamen generiert, mit dem "Datei"-Node.
- Bei einigen anderen Nodes, wie z.B. XML-Attributen, Datenbankfeldern oder der Ausgabekomponente innerhalb einer benutzerdefinierten Funktion, ist nur ein einziger Wert zulässig.

Einbeziehen mehrerer Nodes derselben Quellkomponente in den Kontext:

Dies ist in einigen speziellen Fällen nötig und kann mit Hilfe von [Zwischenvariablen](#) erzielt werden.

Kapitel 7

Reihenfolge der Verarbeitung von Mapping-Komponenten

7 Reihenfolge der Verarbeitung von Mapping-Komponenten

MapForce unterstützt Mappings, die aus mehreren Zielkomponenten bestehen. Jede der Zielkomponenten hat eine Vorschau-Schaltfläche, mit der Sie eine Vorschau des Mapping-Ergebnisses für die jeweilige Komponente anzeigen können.

Wenn das Mapping über die Befehlszeile oder anhand von generiertem Code ausgeführt wird, so wird, unabhängig davon, welche Vorschau gerade aktiv ist, das gesamte Mapping ausgeführt und die Ausgabe für jede einzelne Zielkomponente generiert.

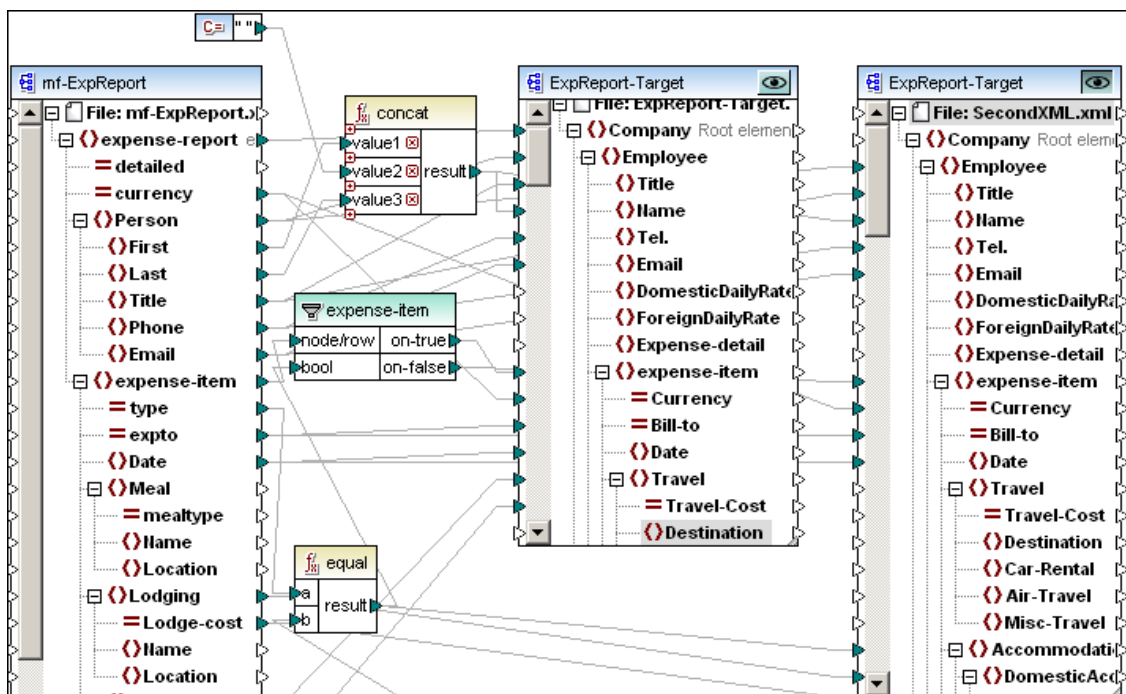
Die Reihenfolge, in der die Zielkomponenten verarbeitet werden, kann durch Ändern der Position der Zielkomponente im Mapping-Fenster beeinflusst werden. Die Position einer Komponente wird anhand ihrer linken oberen Ecke definiert.

Die Zielkomponenten werden anhand ihrer Y-X-Position am Bildschirm von oben nach unten und von links nach rechts verarbeitet.

- Wenn zwei Komponenten dieselbe vertikale Position haben, so wird zuerst die am weitesten links liegende Komponente verarbeitet.
- Wenn zwei Komponenten dieselbe horizontale Position haben, so wird die weiter oben liegende Komponente zuerst verarbeitet.
- In den seltenen Fällen, in denen zwei Komponenten sich an exakt derselben Stelle befinden, wird automatisch eine eindeutige interne Komponenten-ID verwendet. Damit ist eine genau definierte Reihenfolge definiert, die allerdings nicht geändert werden kann.

In der Abbildung unten sehen Sie das Tutorial-Beispiel Tut-ExpReport-multi.mfd aus dem Ordner ...MapForceExamples\Tutorial.

Beide Zielkomponenten (ExpReport-Target) nehmen dieselbe **vertikale** Position ein. Die Vorschau-Schaltfläche ist bei der Zielkomponente auf der rechten Seite aktiv.



Nach Auswahl von XSLT2 und Generierung von Code:

- wird die am weitesten links gelegene Zielkomponente zuerst verarbeitet und die Datei ExpReport.xml generiert.
- wird als nächstes die Komponente rechts davon verarbeitet und die Datei SecondXML.xml generiert.

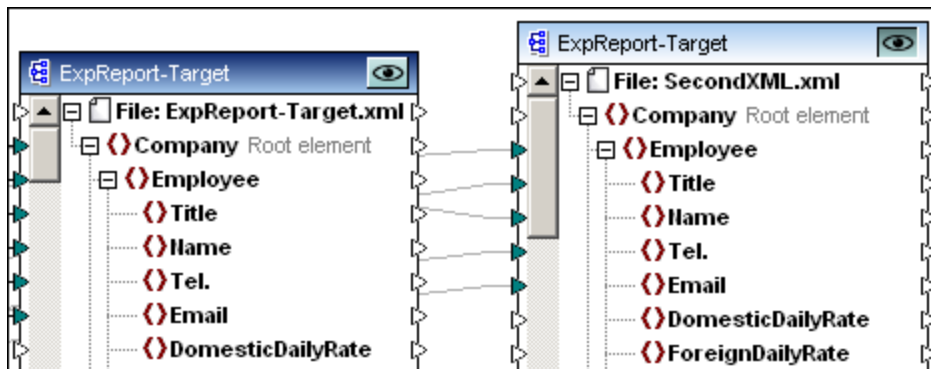
Um dies zu überprüfen, öffnen Sie die Datei **DoTransform.bat** (in dem von Ihnen definierten Ausgabeordner) und sehen Sie, in welcher Reihenfolge die Ausgabedateien generiert werden. ExpReport-Target.xml ist die erste Ausgabedatei, die von der Batch-Datei generiert wird, die zweite Datei ist SecondXML.xml.

```
@echo off

AltovaXML /xslt2 "MappingMapToExpReport-Target.xslt" /in "C:/Documents and Se
Documents/Altova/MapForce2012/MapForceExamples/Tutorial/ExpReport-Target.xml"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
AltovaXML /xslt2 "MappingMapToExpReport-Target2.xslt" /in "C:/Documents and S
Documents/Altova/MapForce2012/MapForceExamples/Tutorial/SecondXML.xml" %*
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

Ändern der Reihenfolge der Mapping-Verarbeitung:

1. Klicken Sie auf die linke Zielkomponente und ziehen Sie sie tiefer als die rechte Komponente.



2. Generieren Sie Ihren Code erneut und werfen Sie einen Blick in die Datei DoTransform.bat.

```
Listr - [c:\temp2\DoTransform.bat]
File Edit Options Help
@echo off

AltovaXML /xslt2 "MappingMapToExpReport-Target.xslt" /in "C:/Documents and Se
Documents/Altova/MapForce2012/MapForceExamples/Tutorial/SecondXML.xml" %*
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
AltovaXML /xslt2 "MappingMapToExpReport-Target2.xslt" /in "C:/Documents and S
Documents/Altova/MapForce2012/MapForceExamples/Tutorial/ExpReport-Target.xml"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

Nun generiert die Batch-Datei als erstes die Datei SecondXML.xml und anschließend die Datei ExpReport-Target.xml.

Verkettete Mappings

Dieselbe oben beschriebene Verarbeitungsreihenfolge gilt auch für [verkettete Mappings](#). Die verkettete Mapping-Gruppe wird jedoch als eine Einheit behandelt. Wenn Sie die Zwischen-, End- oder Zielkomponente eines einzigen verketteten Mappings verschieben, hat dies keine Auswirkung auf die Verarbeitungsreihenfolge.

Die Position der **Endkomponenten** von einzelnen Gruppen hat nur dann einen Einfluss auf die Verarbeitungsreihenfolge, wenn mehrere Ketten bzw. mehrere Zielkomponenten in einem Mapping vorhanden sind.

- Wenn zwei Endkomponenten dieselbe vertikale Position einnehmen, so wird zuerst die weiter links gelegene verarbeitet.
- Wenn zwei Endkomponenten dieselbe horizontale Position einnehmen, so wird zuerst die weiter oben gelegene verarbeitet.
- In den seltenen Fällen, in denen zwei Komponenten sich an exakt derselben Stelle befinden, wird automatisch eine eindeutige interne Komponenten-ID verwendet. Damit ist eine genau definierte Reihenfolge definiert, die allerdings nicht geändert werden kann.

Kapitel 8

Built-In-Ausführungsprozessor

8 Built-In-Ausführungsprozessor

Mit Hilfe des Built-In Ausführungsprozessors können Sie das Ergebnis einer Transformation sofort in der Vorschau anzeigen und speichern, ohne dass vorher Programmcode generiert, kompiliert und anschließend die Ergebnisse angezeigt werden müssen.

Wenn Sie auf die Schaltfläche **BUILTIN**  klicken, wird der Built-In-Ausführungsprozessor explizit als Ausgabeziel ausgewählt. Mit Hilfe des Built-In-Ausführungsprozessors wird auch die Vorschau des Ergebnisses angezeigt, wenn eine der Programmiersprachen (Java, C#, C++) ausgewählt ist.

MapForce lässt sich auch über die [Befehlszeile](#) starten und kann mit Hilfe des Parameters /BUILTIN auch ohne die Erzeugung von Zwischencode ein Ergebnis produzieren.

Built-In-Ausführungsprozessor - Streaming:

Der Built-In-Ausführungsprozessor ist das einzige Ziel, das sowohl das Streaming von Input- als auch Output-XML-, CSV- und FLF-Dateien unterstützt. Dies bedeutet, dass es für die Input- und Output-Dateien keine Größenbeschränkung gibt. Bei sehr großen Output-Dateien wird am unteren Rand des [Ausgaberegisters](#) eine Schaltfläche "Mehr laden.." angezeigt. Wenn Sie darauf klicken, wird ein großer Abschnitt an die bereits sichtbaren Daten angehängt.

Die Schaltfläche "Pretty-Print" wird aktiv, sobald die komplette Datei in das Ausgaberegister geladen wurde.

Digitale Signaturen:

MapForce unterstützt die Erstellung von digitalen XML-Signaturen für XML- und XBRL-Ausgabedateien. Digitale Signaturen können nur erstellt werden, wenn das BUILTIN das Ausgabeziel ist. Eine Signatur wird erstellt, wenn Sie auf die Schaltfläche "Ausgabe" klicken und die Ergebnisdatei gespeichert wird.

Digitale XML-Signaturen sind eine W3C-Spezifikation zum digitalen Signieren eines XML-Dokuments mit einem verschlüsselten Code, anhand dessen überprüft werden kann, ob das XML-Dokument unverändert geblieben ist. Die XML-Signaturfunktion in MapForce unterstützt nur Zertifikate vom Typ RSA-SHA1 und DSA-SHA1.

Nähere Informationen dazu finden Sie unter [Digitale Signaturen](#).

Kapitel 9

Globale Ressourcen

9 Globale Ressourcen

Globale Ressourcen stellen eine große Verbesserung hinsichtlich der Interoperabilität zwischen den Produkten der Altova-Produktlinie dar und stehen derzeit in den folgenden Altova-Produkten zur Verfügung: XMLSpy, MapForce, StyleVision und DatabaseSpy. Benutzer der Altova MissionKit-Pakete haben in den entsprechenden Einzelprodukten Zugriff auf dieselben Funktionalitäten.

Allgemeine Verwendungsweise:

- Es können Arbeitsabläufe definiert werden, bei denen verschiedene Altova-Applikationen zur Verarbeitung der Daten verwendet werden.
- Eine Applikation kann eine Zielapplikation aufrufen, die Datenverarbeitung dort starten und die Daten wieder an die ursprüngliche Applikation zurückleiten.
- Definition von Input- und Output-Daten, Dateipfaden sowie Datenbanken als globale Ressourcen.
- Wechseln zwischen globalen Ressourcen zur Laufzeit, um zwischen Ressourcen für Entwicklungszwecke und solchen für die reale Anwendung zu wechseln.
- Was wäre wenn-Szenarios für Zwecke der Qualitätskontrolle.

Der Standardpfad der Definitionsdatei für globale Ressourcen, **GlobalResources.xml**, lautet c:\Documents and Settings\UserName\My Documents\Altova\.. Dies ist der Standardpfad für alle Altova-Applikationen, die globale Ressourcen verwenden können. Änderungen, die an globalen Ressourcen vorgenommen werden, stehen daher automatisch in allen Applikationen zur Verfügung. Der Pfad und Name der Datei kann geändert werden. Nähere Informationen dazu siehe [Globale Ressourcen - Eigenschaften](#).

Allgemeine Funktionsweise:

- Globale Ressourcen werden in einer Applikation [definiert](#) und automatisch gespeichert.
- Globale Ressourcen werden Komponenten [zugewiesen](#), deren Daten variabel sein sollen.
- Die globale Ressource wird in einer Applikation aufgerufen / [aktiviert](#), sodass Sie zur Laufzeit Ressourcen wechseln können.

In diesem Abschnitt lesen Sie, wie globale Ressourcen mittels bestehender Mappings aus dem Ordner ...MapForceExamples\Tutorial\ definiert und verwendet werden können.

So aktivieren Sie die Symbolleiste "Globale Ressourcen":

Wählen Sie den Menübefehl **Extras | Anpassen |** klicken Sie auf das Register **Symbolleiste** und aktivieren Sie das Kontrollkästchen "Globale Ressourcen". Daraufhin wird die Symbolleiste "Globale Ressourcen" angezeigt.



Über die Auswahlliste können Sie zwischen den verschiedenen Ressourcen wechseln, der Eintrag "Default" ist immer verfügbar.

Wenn Sie auf die Schaltfläche "Globale Ressourcen"  klicken, wird das Dialogfeld "Globale Ressourcen" geöffnet (alternativ dazu Extras | Globale Ressourcen).

9.1 Globale Ressourcen - Dateien

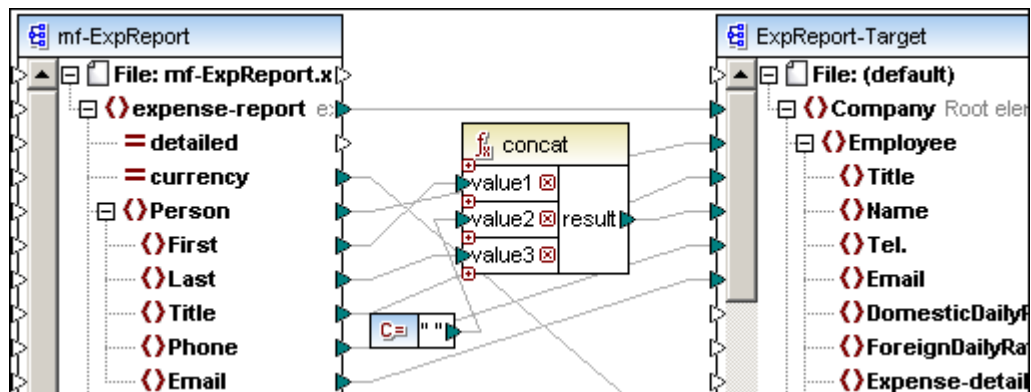
Globale Ressourcen in MapForce:

- Es kann jede Input/Output-Datei als globale Ressource definiert werden, z.B. XML, XML Schema, Text/CSV-, Datenbanken, EDI- und Excel 2007 (und höher)- Dateien.

Ziel dieses Abschnitts:

- die Input-Datei für die Quellkomponente, **mf-ExpReport**, zu einer globalen Ressource zu machen.
- zur Laufzeit zwischen den beiden XML-Dateien, die die Input-Daten der Quellkomponente bereitstellen, zu wechseln und die erzeugte XML-Ausgabe auf dem Register Ausgabevorschau anzuzeigen.

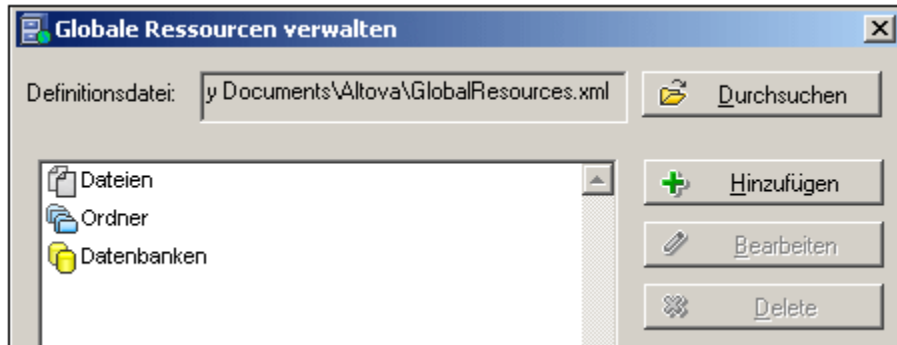
In diesem Abschnitt wird die Datei **Tut-ExpReport.mfd** aus dem Ordner ... \MapForceExamples\Tutorial\ verwendet.



9.1.1 Definieren / Hinzufügen einer globalen Ressource

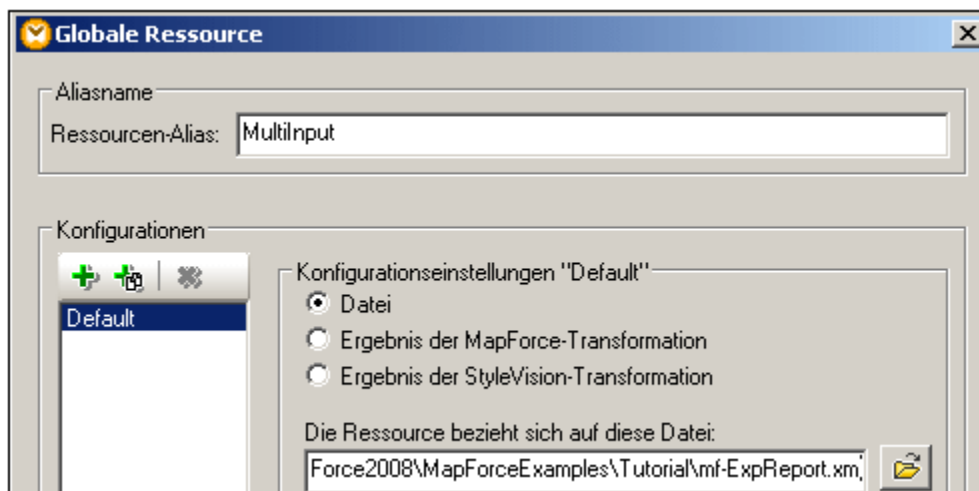
Definieren / Hinzufügen einer globalen Ressourcendatei

1. Klicken Sie auf die Schaltfläche "Globale Ressource" , um das Dialogfeld zu öffnen.

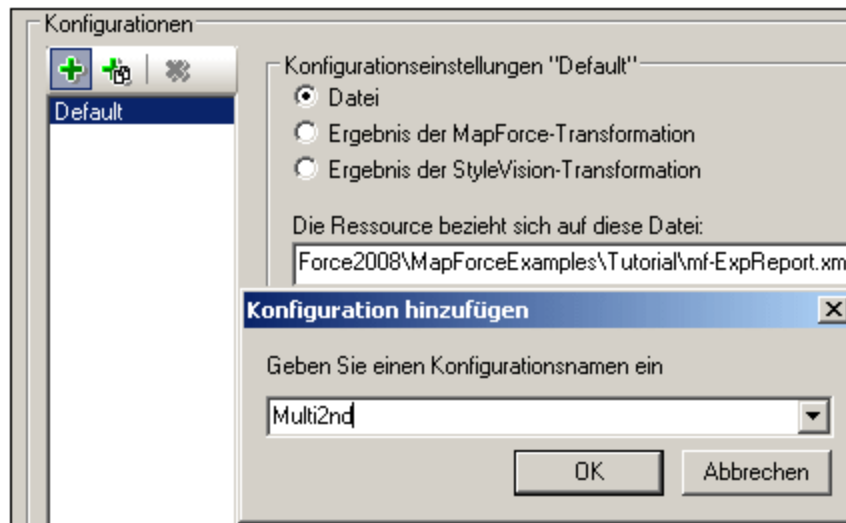


Hier sehen Sie eine leere globale Ressourcendatei.

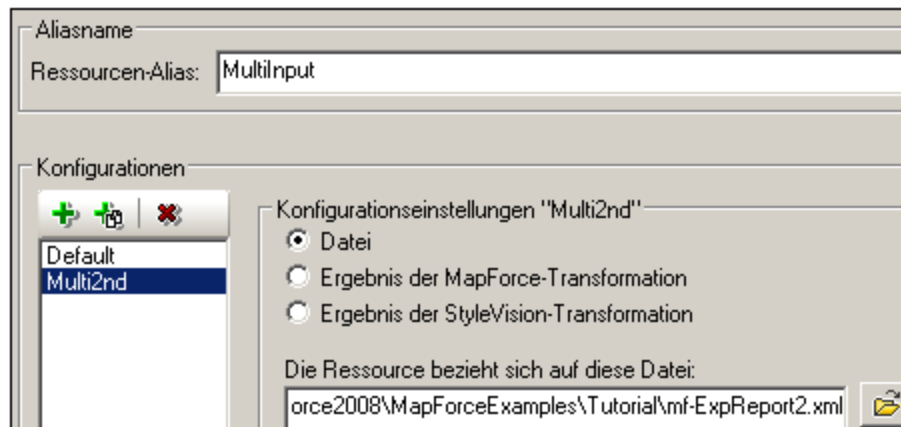
2. Klicken Sie auf die Schaltfläche **Hinzufügen** und wählen Sie im Popup-Fenster den Eintrag **Datei**.
3. Geben Sie den Namen des Ressourcenalias ein, z.B. **MultilInput**.
4. Klicken Sie auf das Symbol zum Öffnen des Ordners und wählen Sie die XML-Datei aus, die als Standard (Default) Input-Datei dienen soll, z.B: **mf-ExpReport.xml**.



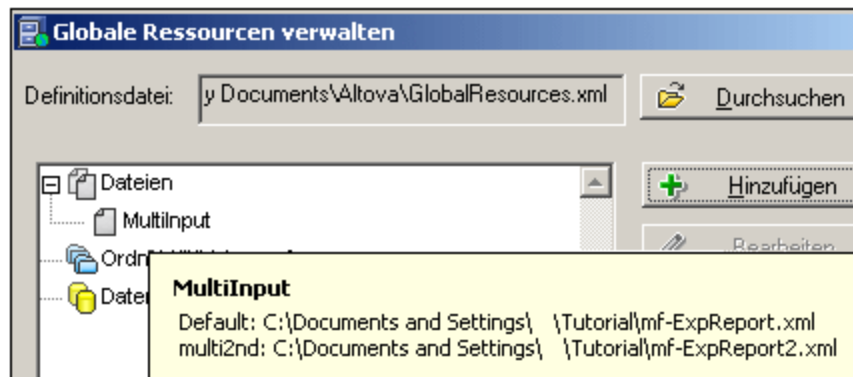
5. Klicken Sie in der Gruppe **Konfigurationen** auf die Schaltfläche "Hinzufügen" , um eine neue Konfiguration zum aktuellen Alias hinzuzufügen. Beachten Sie, dass Sie mit der Schaltfläche "Konfiguration kopieren"  eine ausgewählte Konfiguration kopieren und unter einem neuen Namen speichern können.



6. Geben Sie einen Namen für die Konfiguration ein, **Multi2nd**, und klicken Sie zur Bestätigung auf OK.
Multi2nd wurde nun zur Liste der Konfigurationen hinzugefügt.
7. Klicken Sie nochmals auf die Schaltfläche "Ordner öffnen" und wählen Sie die XML-Datei aus, die als Input-Datei für die Konfiguration multi2nd dienen soll, z.B. **mf-ExpReport2.xml**.



8. Klicken Sie auf OK, um die Definition der Ressource abzuschließen.
Der Alias "MultiInput" wurde nun zum Abschnitt "Dateien" der globalen Ressourcen hinzugefügt.
Wenn Sie den Mauszeiger über einen Aliaseintrag platzieren, wird ein Tooltip mit der Definition geöffnet.



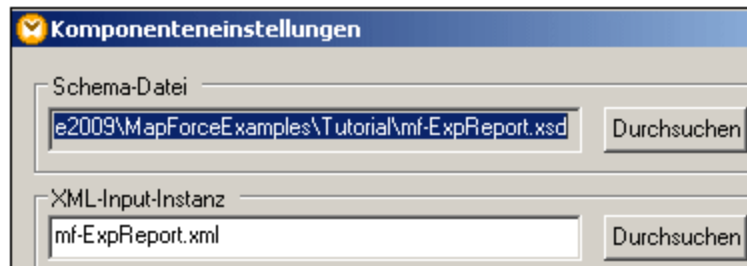
9. Klicken Sie zur Bestätigung auf OK.
Damit ist die globale Ressource nun fertig definiert. Im nächsten Schritt wird [eine globale Ressource nun einer Komponente zugewiesen](#).

9.1.2 Zuweisen einer globalen Ressource

Zuweisen einer globalen Ressource zu einer Komponente

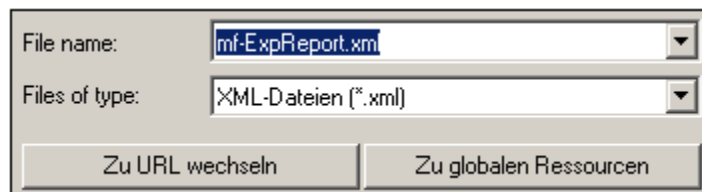
Wir müssen die globale Ressource nun der Komponente zuweisen, für die sie verwendet werden soll.

1. Doppelklicken Sie auf die Komponente **mf-ExpReport** und klicken Sie auf die Schaltfläche "Durchsuchen" neben dem Feld "XML-Input-Instanz".

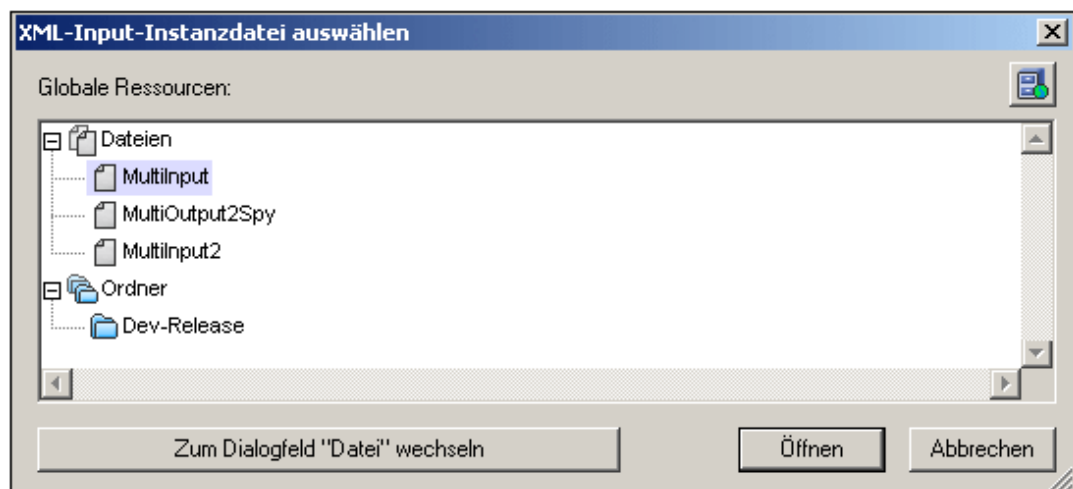


Daraufhin wird das Dialogfeld "XML-Input-Instanzdatei auswählen" geöffnet.

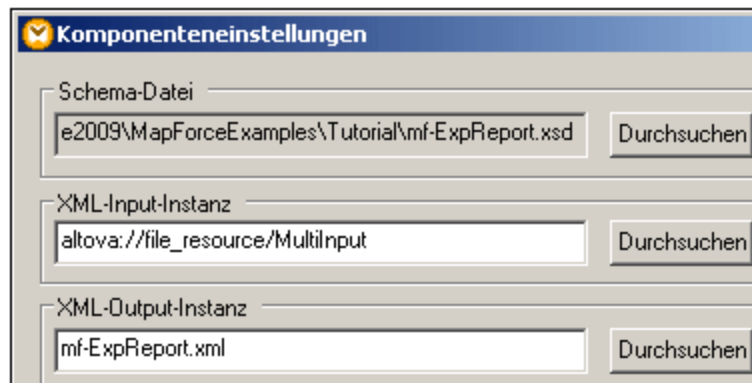
2. Klicken Sie auf die Schaltfläche **Zu globalen Ressourcen** am unteren Rand des Dialogfelds.



3. Klicken Sie auf die Ressource, die Sie zuweisen möchten, in diesem Fall auf **Multilnput**, und klicken Sie auf OK.



Die **XML-Input-Instanzdatei** enthält nun eine Referenz auf eine Ressource, also. **altova://file_resource/Multilnput**.



4. Klicken Sie auf OK, um die Zuweisung einer Ressource zu einer Komponente fertig zu stellen.
Der nächste Schritt ist die [Verwendung / Aktivierung einer globalen Ressource](#).

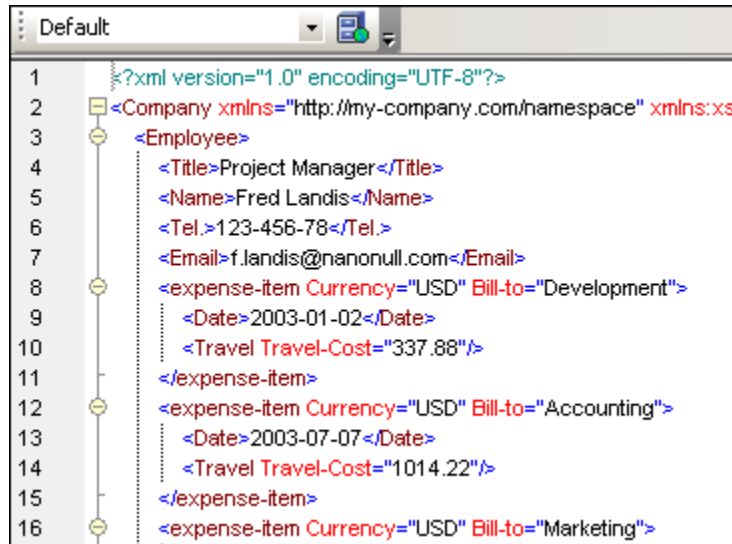
9.1.3 Verwendung / Aktivierung einer globalen Komponente

Verwenden / Aktivieren einer globalen Ressource

Zur Zeit ist die vorhin definierte Konfiguration **Default** für den Alias **MultiInput** aktiv. Dies sehen Sie daran, dass der Eintrag in der Symbolleiste "Globale Ressourcen" "Default" lautet.



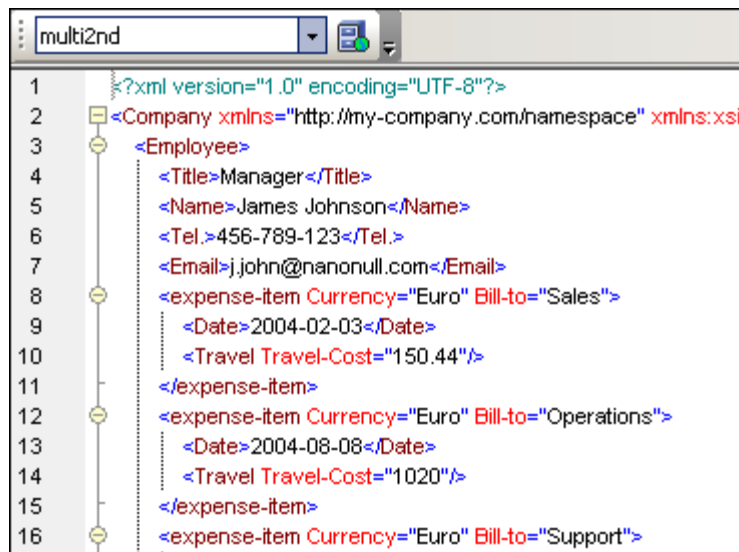
1. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



2. Klicken Sie auf das Mapping-Register, um zur Mapping-Ansicht zurückzukehren.
3. Klicken Sie auf die Auswahlliste "Globale Ressourcen" und wählen Sie darin den Eintrag **multi2nd**.



4. Klicken Sie auf die Schaltfläche "Ausgabe", um das neue Ergebnis zu sehen. Die Datei mf-ExpReport2.xml wird nun als Quellkomponente für das Mapping verwendet und erzeugt nun eine andere Ausgabe.

**Anmerkung:**

Die **aktuell aktive** globale Ressource (**multi2nd** in der Symbolleiste "Globale Ressourcen") bestimmt das Ergebnis des Mappings. Dies ist auch der Fall, wenn Sie Code generieren.

9.2 Globale Ressourcen - Ordner

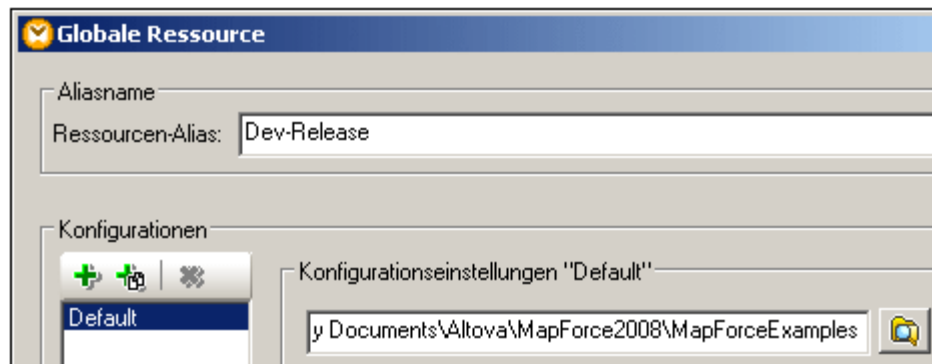
Auch Ordner können als globale Ressource definiert werden, d.h. in diesem Fall, dass Input-Komponenten Dateien enthalten können, die sich auf andere Ordner beziehen. Dies kann z.B. für Entwicklungs- und Release-Zyklen verwendet werden.

Ordner für Output-Komponenten in MapForce zu definieren, hat wenig Sinn, da Sie beim Generieren einer XSLT-Datei oder bei der Codegenerierung für andere Programmiersprachen immer nach den Zielordnern gefragt werden.

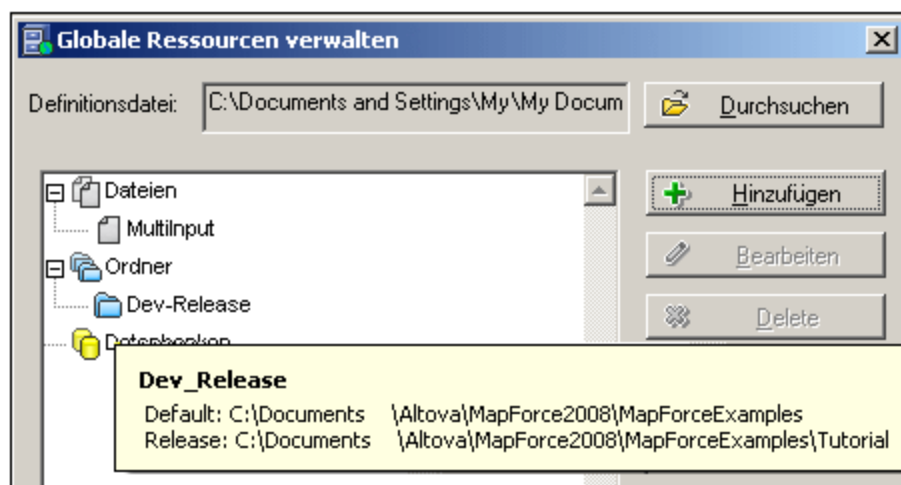
Die in diesem Abschnitt verwendete Mapping-Datei finden Sie unter dem Namen "**global-folder.mfd**" im Ordner ...\\MapForceExamples\\Tutorial\\.

Definieren / Hinzufügen von Ordnern für globale Ressourcen

1. Klicken Sie auf die Schaltfläche "Globale Ressource" , um das Dialogfeld zu öffnen.
2. Klicken Sie auf die Schaltfläche **Hinzufügen** und wählen Sie im Popup-Fenster den Eintrag **Ordner**.
3. Geben Sie den Namen des Ressourcenalias ein, z.B. **Dev_Release**.
4. Klicken Sie auf die Schaltfläche "Ordner öffnen" und wählen Sie den Standard-Input-Ordner "Default", ...\\MapForceExamples.



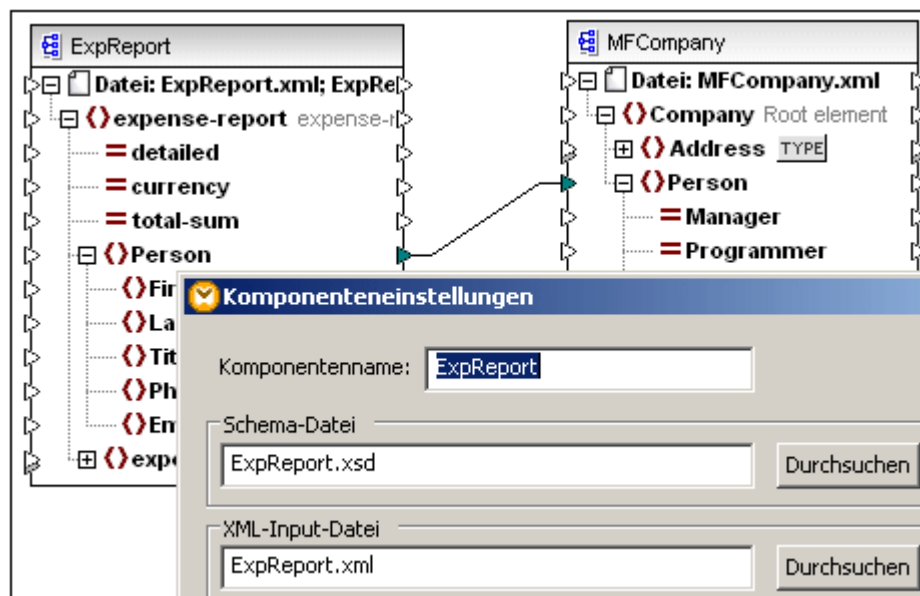
5. Klicken Sie in der Gruppe "Konfigurationen" auf die Schaltfläche "Hinzufügen" , um eine neue Konfiguration zum aktuellen Alias hinzuzufügen und geben Sie einen Namen dafür ein, z.B. Release. Beachten Sie, dass Sie mit der Schaltfläche "Konfiguration kopieren"  eine ausgewählte Konfiguration kopieren und unter einem neuen Namen speichern können.
6. Klicken Sie auf die Schaltfläche "Ordner öffnen" und wählen Sie den Release Input-Ordner, ...\\MapForceExamples\\Tutorial.



7. Klicken Sie auf OK, um die Definition des globalen Ordners abzuschließen.

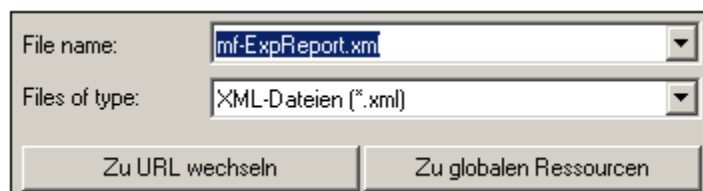
Zuweisen des Ordners für globale Ressourcen:

1. Doppelklicken Sie auf die Komponente **ExpReport** und klicken Sie auf die Schaltfläche "Durchsuchen" neben dem Feld **XML-Input-Datei**.

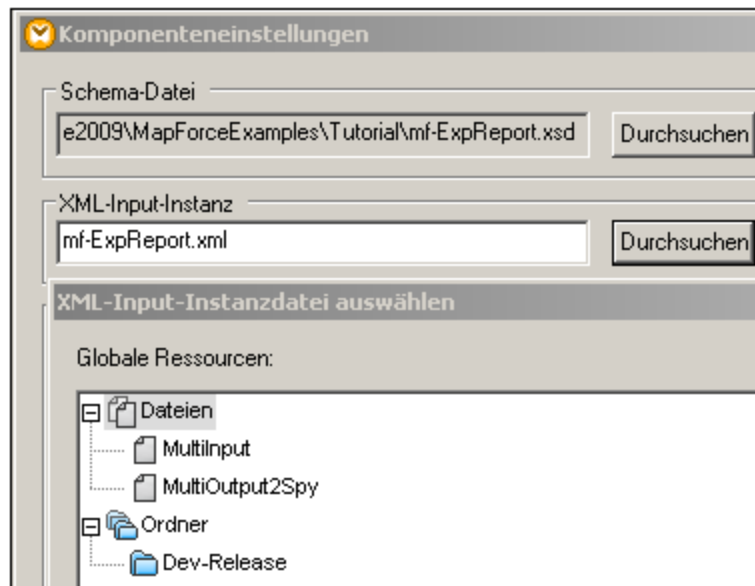


Daraufhin wird das Dialogfeld "XML-Input-Instanzdatei auswählen" geöffnet.

2. Klicken Sie auf die Schaltfläche **Zu globalen Ressourcen** am unteren Rand des Dialogfelds.

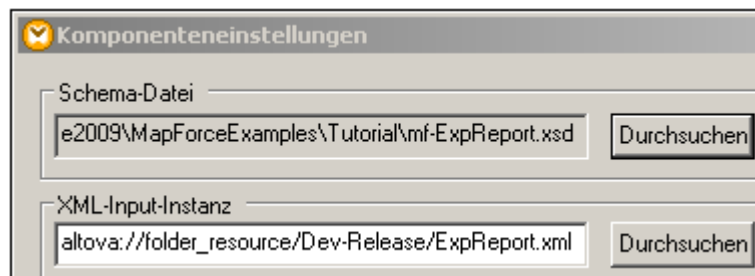


3. Klicken Sie auf die gewünschte Ressource, in diesem Fall auf **Dev_Release**, und bestätigen Sie mit OK.



Daraufhin erscheint das Dialogfeld "....öffnen".

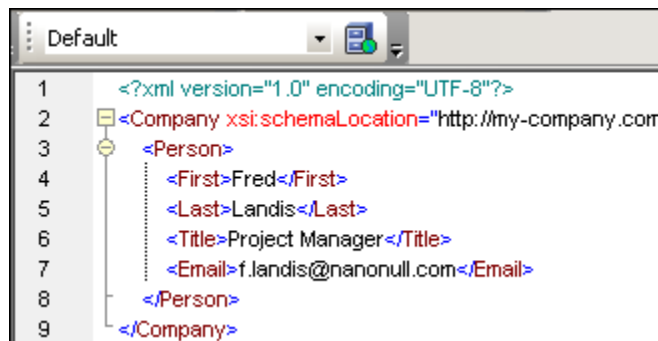
4. Wählen Sie in jedem der Ordner den Dateinamen, der sowohl als **Ressourcendatei** für die Entwicklung als auch für die Release verwendet werden soll, z.B. **ExpReport.xml** und klicken Sie auf OK, um die Zuweisung des Resourceordners abzuschließen.



Beachten Sie, dass die Datei in beiden Verzeichnissen vorhanden ist, aber einen unterschiedlichen Inhalt hat.

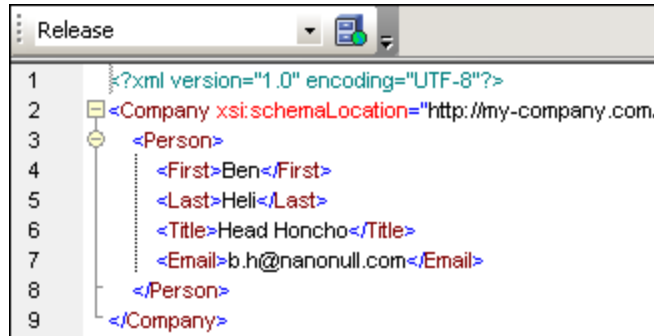
Wechseln des Resourceordners zur Laufzeit:

1. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis der Transformation zu sehen.
Beachten Sie, dass dies die Konfiguration / des Ordners **Default** in .../MapforceExamples ist.



2. Klicken Sie auf das Mapping-Register, um zum Mapping-Fenster zurückzukehren.

3. Klicken Sie auf die Auswahlliste "Globale Ressource" und wählen Sie den Eintrag "**Release**".
3. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis bei Verwendung der Globalen Ressourcen "Release" zu sehen.

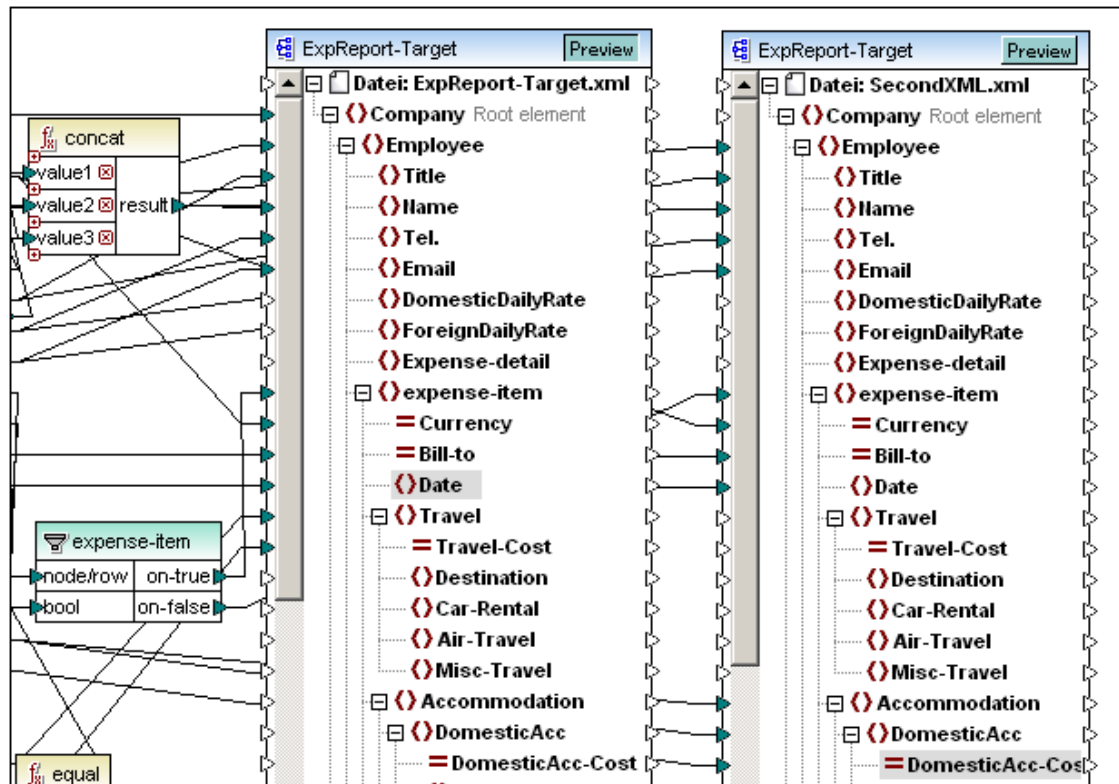


Es wird nun die Ausgabe für den Ordner "Release" aus .../MapforceExamples/Tutorial angezeigt.

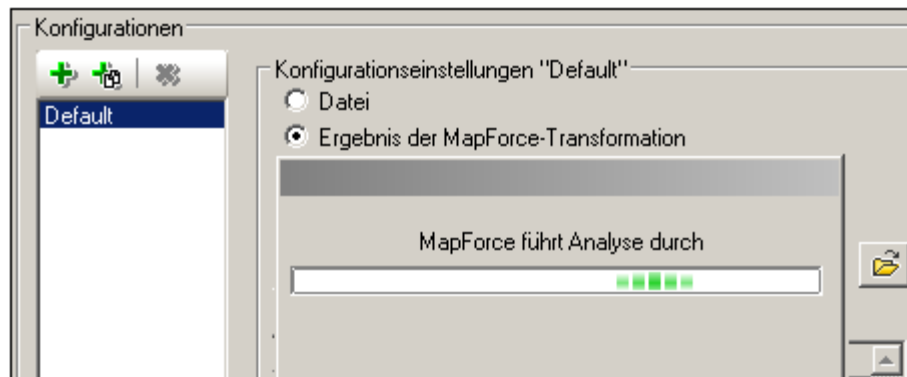
9.3 Globale Ressourcen - Arbeitsablauf in Applikationen

Ziel dieses Abschnittes ist ein ArbeitsablaufszENARIO zwischen zwei Altova-Applikationen zu erstellen. Der Arbeitsablauf beginnt in XMLSpy, das MapForce startet und von wo aus die generierte XML-Ausgabedatei zur weiteren Verarbeitung wieder an XMLSpy übergeben wird.

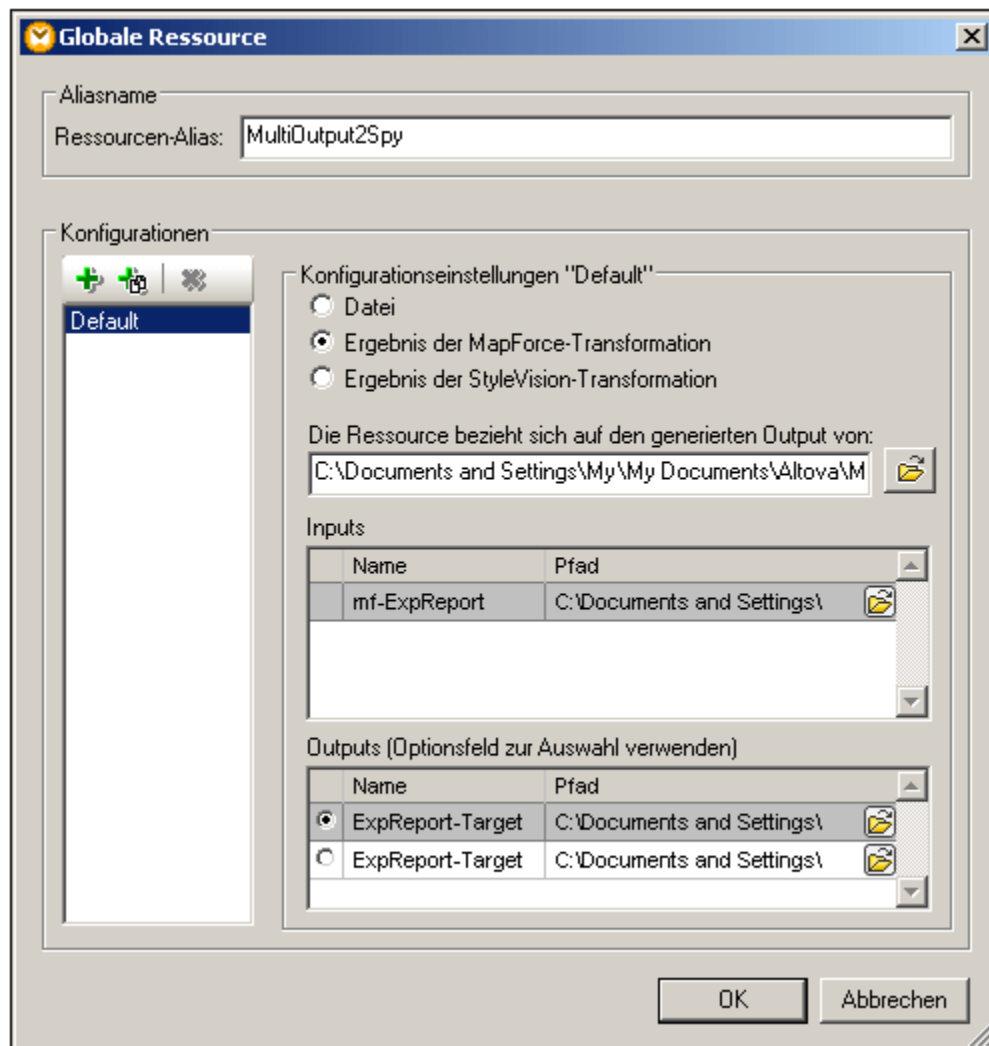
In diesem Mapping werden zwei Output-Komponenten verwendet, um zwei Arten einer gefilterten Ausgabe zu erzeugen; Reisekosten (Travel expenses) und andere Ausgaben (Non-travel expenses) der Reisekosten-Input-Datei. In diesem Abschnitt wird die Mapping-Datei **Tut-ExpReport-multi.mfd** aus dem Ordner ...\\MapForceExamples\\Tutorial\\ verwendet.



1. Klicken Sie auf die Schaltfläche "Globale Ressource" , um das Dialogfeld zu öffnen.
2. Klicken Sie auf die Schaltfläche "Hinzufügen" und wählen Sie im Popup-Fenster den Eintrag "Datei".
3. Geben Sie den Namen des Resourcealias ein, z.B. **MultiOutput2Spy**
4. Klicken Sie auf das Optionsfeld **"Ergebnis der MapForce-Transformation"** und klicken Sie auf die Schaltfläche "Datei öffnen".
5. Wählen Sie das Mapping **Tut-ExpReport-multi.mfd** aus.

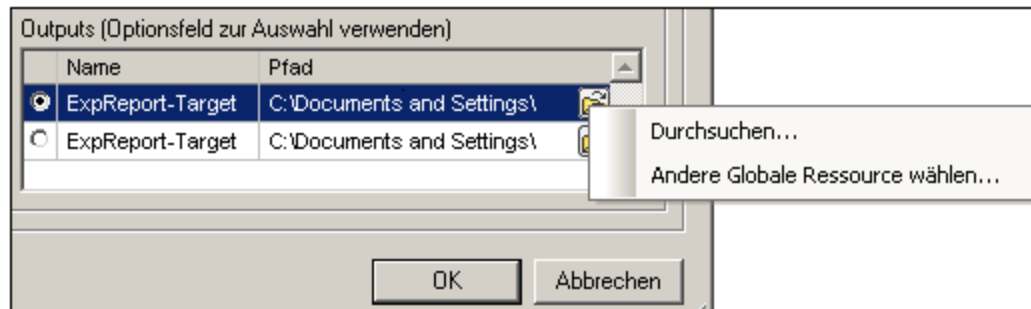


MapForce analysiert das Mapping und zeigt die Input- und Output-Dateien in separaten Listenfeldern an.

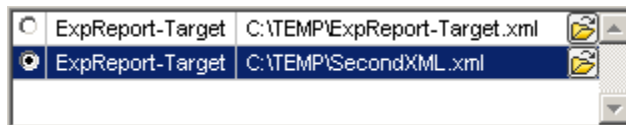


6. Klicken Sie auf den Optionsfeldeintrag im Abschnitt **Outputs**, falls er noch nicht ausgewählt ist.
Beachten Sie, dass der Name der Ausgabedatei **ExpReport-Target.xml** ist, und dass wir zur Zeit die Standardkonfiguration (Default) definieren.
7. Klicken Sie auf die Schaltfläche , um einen **neuen** Pfad für die Ausgabedatei zu

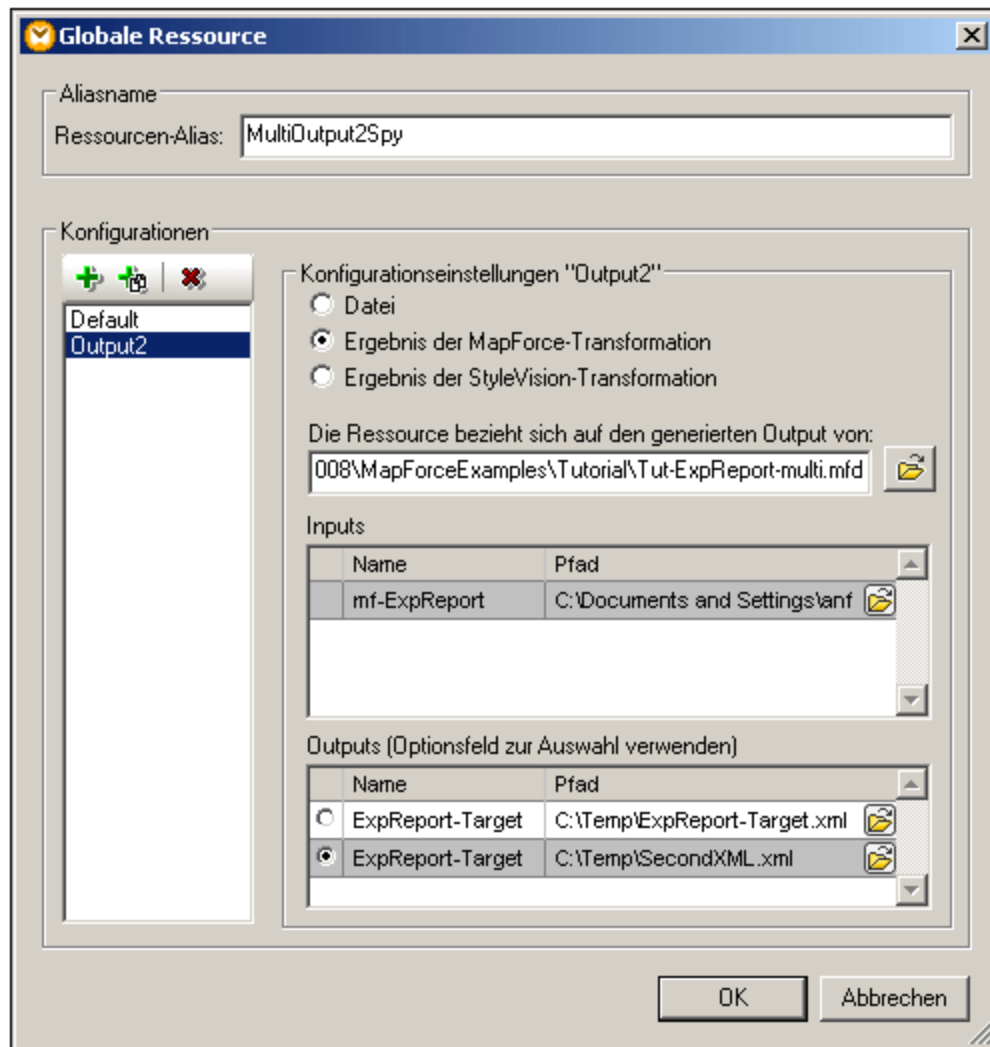
definieren und wählen Sie im Popup-Menü **Durchsuchen....**



8. Geben Sie den neuen Ausgabepfad ein, z.B. C:\Temp und klicken Sie auf die Schaltfläche "Speichern". Dieser Pfad kann sich von dem in den Komponenteneinstellungen definierten unterscheiden.
9. Klicken Sie in der Gruppe "Konfigurationen" auf die Schaltfläche "Hinzufügen"  (dieses Dialogfelds) um eine neue Konfiguration zum Ressourcenalias hinzuzufügen.
10. Geben Sie den Namen der Konfiguration ein, z.B. **Output2**, klicken Sie auf das Symbol "Datei öffnen" und aktivieren Sie das untere Optionsfeld im Listefeld "Outputs". Beachten Sie, dass der Name der Ausgabedatei **SecondXML.xml** ist.



11. Klicken Sie auf die Schaltfläche "Speichern unter" , um den neuen Pfad der Ausgabedatei zu speichern, z.B. C:\Temp.
Anmerkung: Wenn Sie im Popup-Fenster auf die Schaltfläche "**Andere Globale Ressource wählen...**" klicken, können Sie die MapForce-Ausgabe als globale Ressource definieren, d.h. die Ausgabe wird in einer Datei gespeichert, die die globale Ressource referenziert.



12. Klicken Sie auf OK, um die neuen globalen Ressourcen zu speichern.
Der neue Ressourcealias **MultiOutput2Spy** wurde nun zu Definitionsdatei für globale Ressourcen hinzugefügt.

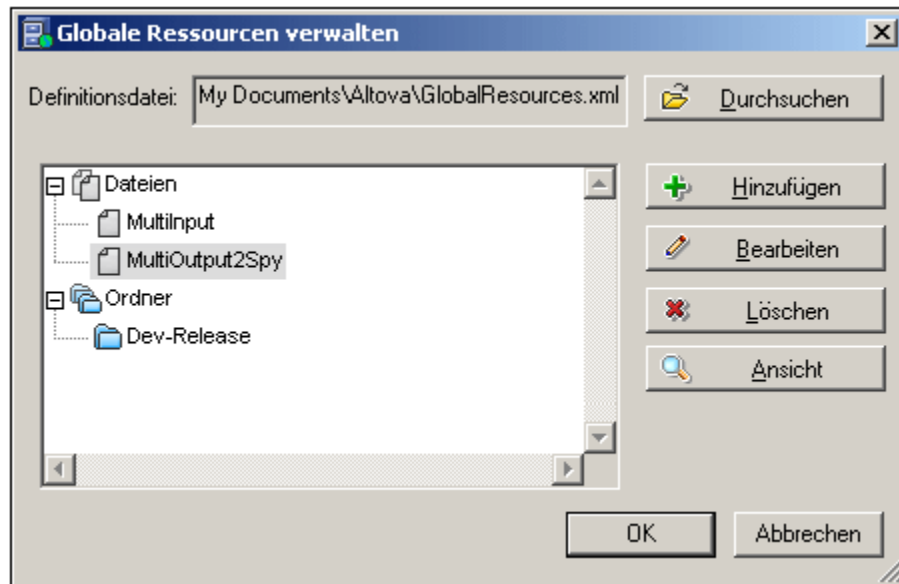


13. Klicken Sie auf OK., um die Definition abzuschließen.

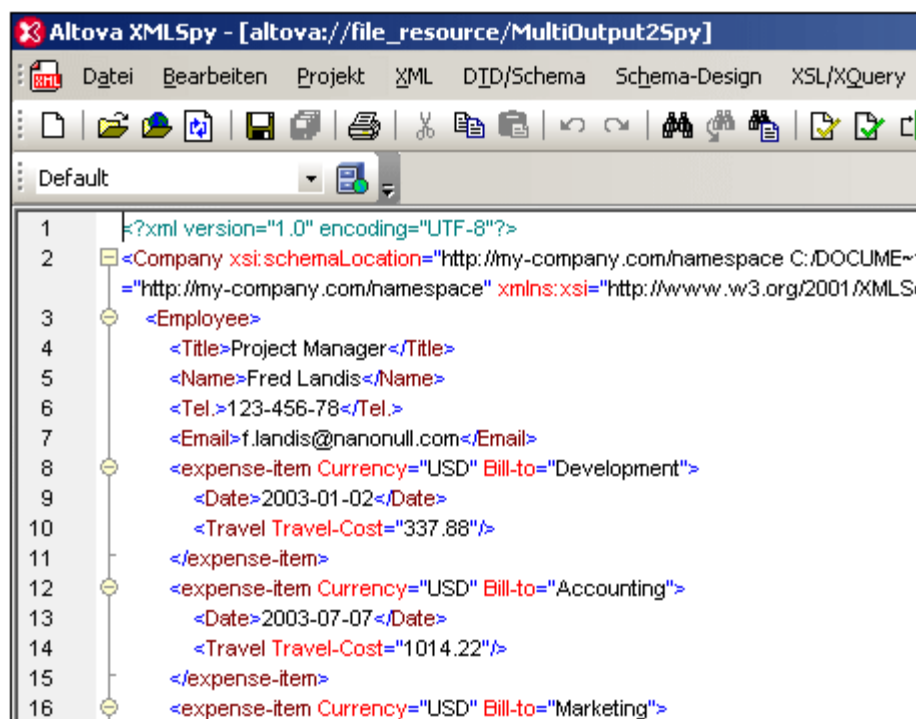
9.3.1 Starten des Applikationsarbeitsablaufs

In diesem Abschnitt wird beschrieben, wie die globale Ressource in XMLSpy aktiviert wird und wie die MapForce Transformation, die daraus resultiert, wieder an XMLSpy zurückgeleitet wird.

1. Starten Sie XMLSpy und schließen Sie MapForce, falls Sie das Programm geöffnet haben, um besser zu sehen, wie die beiden Applikationen interagieren.
2. Wählen Sie in XMLSpy die Menüoption **Extras | Globale Ressourcen**.
3. Wählen Sie den Eintrag **MultiOutput2Spy** aus und klicken Sie auf die Schaltfläche **Ansicht**.



Es erscheint ein Meldungsfeld, in dem Sie darüber informiert werden, dass MapForce eine Transformation durchführt. Das Ergebnis der Transformation wird im Fenster "Textansicht" angezeigt.

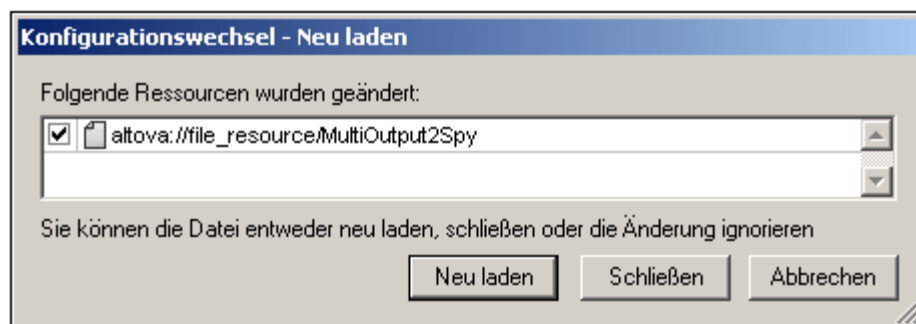


Anmerkung:

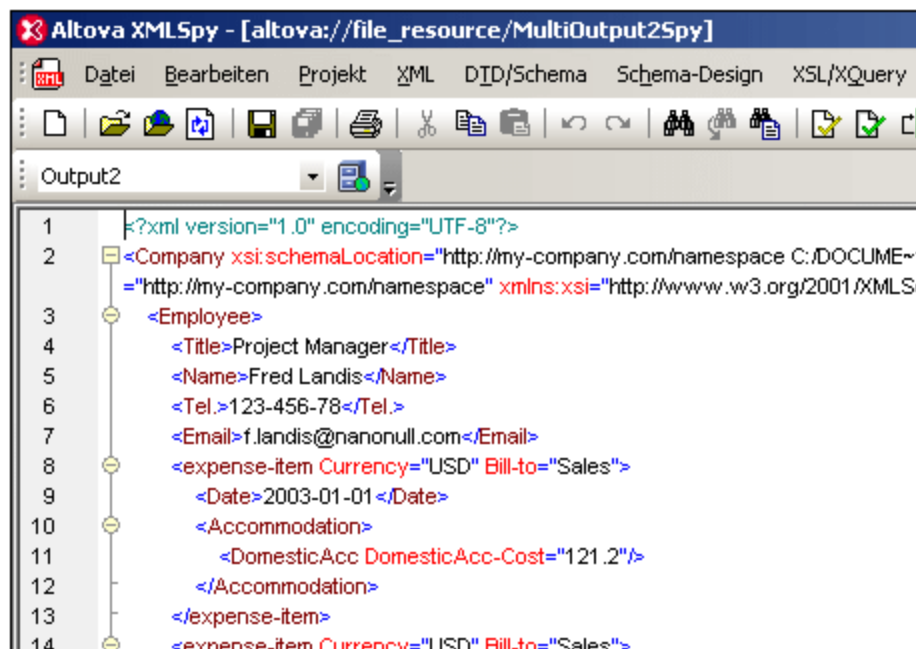
- Die derzeit ausgewählte Konfiguration ist **"Default"**.
- Der Name des Ressourcealias steht in der Applikationstitelleiste **altova://file_resource/MultiOutput2Spy**.
- Die Ausgabedatei wurde als "Untitled1.xml" zur weiteren Verarbeitung geöffnet.
- Die Datei **ExpReport-Target.xml** wurde in den Ordner C:\Temp kopiert.

So rufen Sie die Ausgabe für die anderen Kosten (Nicht-Reisekosten) ab:

1. Klicken Sie auf die Auswahlliste "Globale Ressourcen" und wählen Sie **"Output2"**. Daraufhin erscheint ein entsprechendes Meldungsfeld.



2. Klicken Sie auf **Neu laden**, um die zweite durch die Ressource definierte Ausgabedatei abzurufen.



Das Ergebnis der Transformation wird im Fenster "Textansicht" angezeigt und überschreibt die vorherige Datei MultiOutput2Spy.xml.

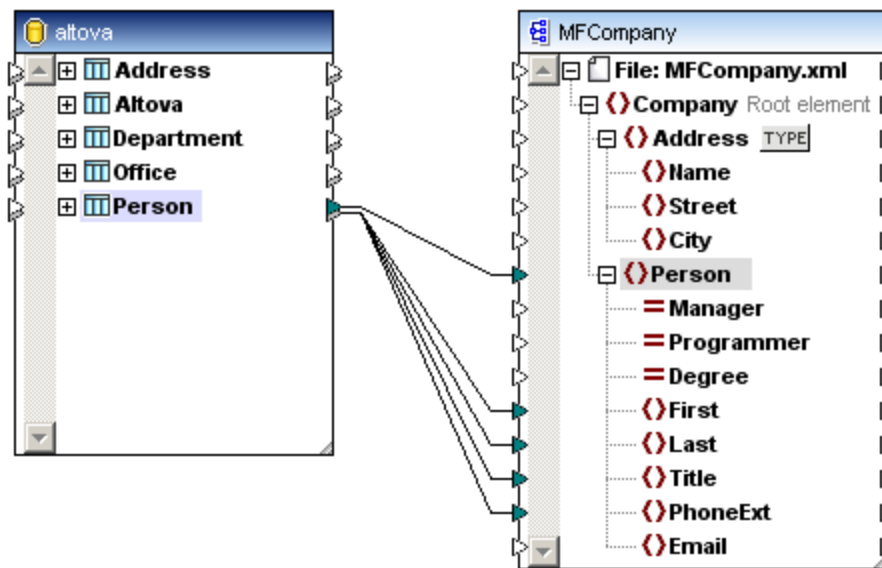
Anmerkung:

- Die aktuell ausgewählte Konfiguration ist "**Output2**".
- Die Ausgabedatei wurde als "Untitled1.xml" für die weitere Verarbeitung geöffnet.
- Die Datei **SecondXML.xml** wurde in den Ordner C:\Temp kopiert.

9.4 Globale Ressourcen - Datenbanken

Auch Datenbankkomponenten können als globale Ressource definiert werden, sodass Sie - z. B. zu Entwicklungs- und Releasezwecken - unterschiedliche Datenbanken referenzieren können. Datenbankressourcen können sowohl als Quell- als auch Zielkomponente verwendet werden.

Die in diesem Abschnitt verwendete Mapping-Datei ist unter dem Namen "**PersonDB.mfd**" im Ordner [...\MapForceExamples\Tutorial\](#) verfügbar.

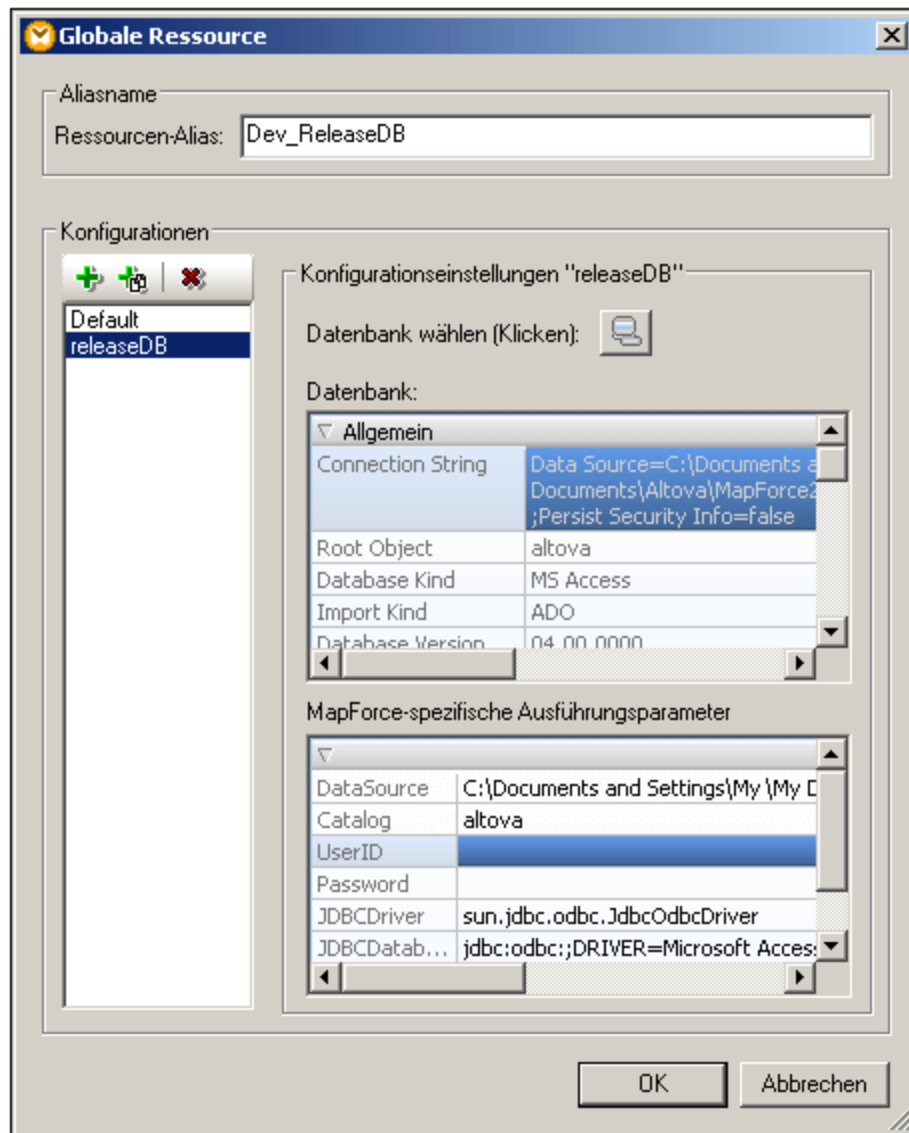


Definieren / Hinzufügen einer globalen Datenbankressource

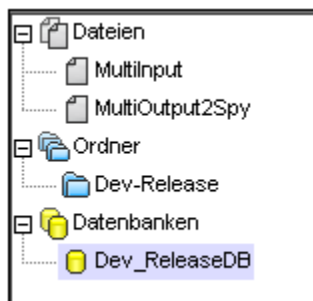
1. Klicken Sie auf die Schaltfläche "Globale Ressource"  um das Dialogfeld zu öffnen.
2. Klicken Sie auf die Schaltfläche **Hinzufügen** und wählen Sie im Popup-Menü den Eintrag **Datenbank**.
3. Geben Sie den Namen des Ressourcealias ein, z.B. **Dev_ReleaseDB**.



4. Klicken Sie auf die Schaltfläche "Verbindungsassistent"  und wählen Sie die Datei Altova.mdb im Ordner [...\MapForceExamples](#). Dies ist die **Entwicklungsdatenbank**.
5. Klicken Sie in der Gruppe **Konfigurationen** auf die Schaltfläche "Hinzufügen" , um eine neue Konfiguration zum aktuellen Alias hinzuzufügen und geben Sie einen Namen dafür ein, z.B. **releaseDB**.



6. Klicken Sie auf die Schaltfläche "Verbindungsassistent"  und wählen Sie die **Releasedatenbank** aus, z.B. altova.mdb im Ordner ...\\MapForceExamples\\Tutorial. Klicken Sie auf OK, um die Definition der Datenbankressource abzuschließen.



Bitte beachten Sie:

Alle Datenbankeinstellungen in den Listenelementen "**Datenbank**" oder "**MapForce Ausführungparameter**" können bearbeitet/geändert werden, sobald Sie eine

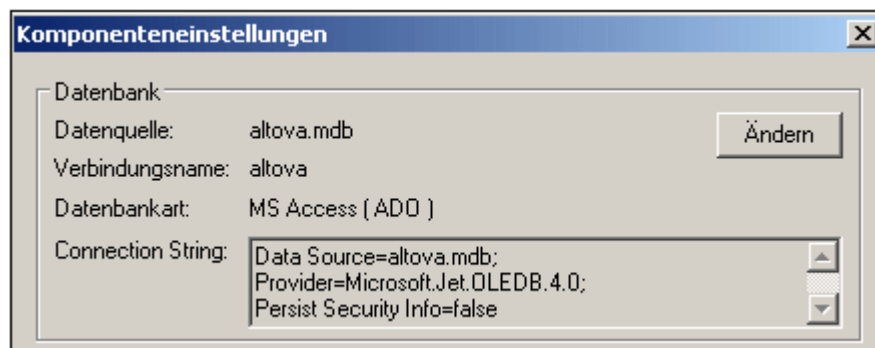
Datenbank ausgewählt haben. Wenn Sie z.B. in das Feld "Passwort" doppelklicken, können Sie das Passwort der aktuellen Datenbank aktualisieren.

Das Listenfeld "MapForce Ausführungsparameter" wird mit den Standardwerten aus der Datenbank befüllt, sobald die Datenbank ausgewählt wurde. Diese Parameter werden vom Built-In-Ausführungsprozessor beim Generieren von Code und der Ausgabevorschau verwendet.

Je nachdem, zu welcher Datenbank eine Verbindung hergestellt wird, wird z.B. bei einer IBM DB2-Datenbank das Dialogfeld "Root-Objekt wählen" angezeigt. Sie können nun entweder über die Schaltfläche "Als Root-Objekt" sofort ein Root-Objekt definieren oder die Auswahl erst später vornehmen, wenn Sie auf die Schaltfläche "Überspringen" klicken.

Zuweisen der Datenbankressource:

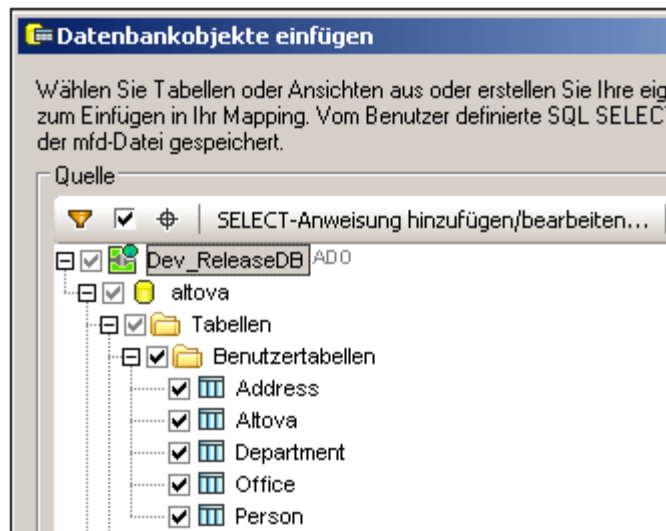
1. Doppelklicken Sie auf die Datenbankkomponente **Altova** und klicken Sie auf die Schaltfläche "Ändern" (und anschließend, falls nötig, auf die Schaltfläche "Globale Ressourcen").



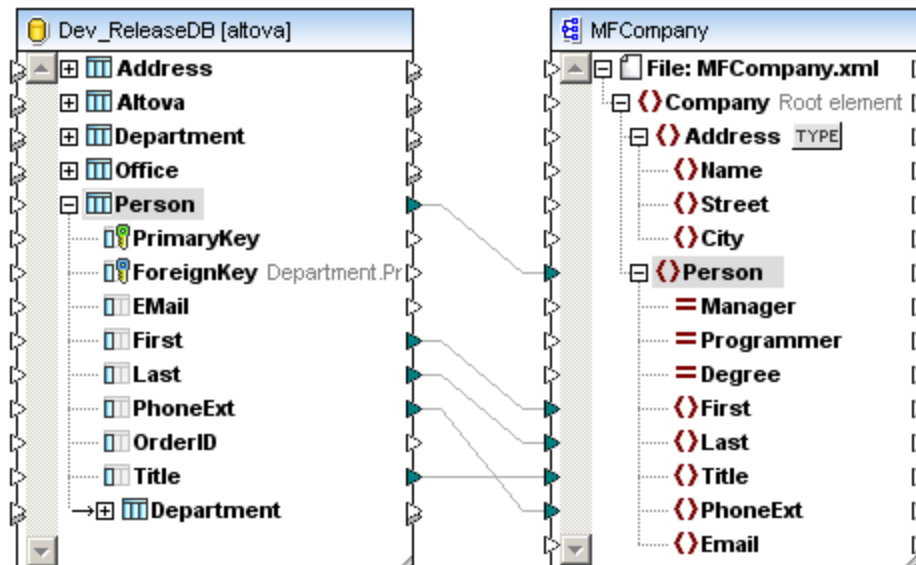
Daraufhin wird das Dialogfeld "Datenbank auswählen" geöffnet.



2. Wählen Sie den Eintrag **Dev_ReleaseDB** und klicken Sie auf die Schaltfläche "Verbindung herstellen".

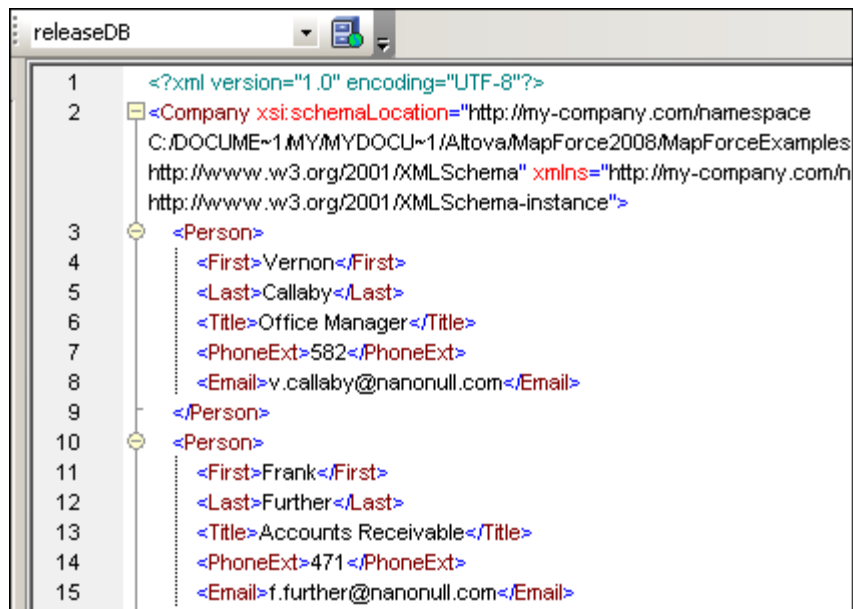


3. Wählen Sie die Tabellen, die in der Komponente zur Verfügung stehen sollen aus, klicken Sie auf die Schaltfläche "Einfügen" und anschließend auf OK. Der Ressourcenalias wird nun in der Komponentenüberschrift Dev_ReleaseDB [Altova] angezeigt.



Wechseln der Datenbankressource zur Laufzeit:

1. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen. Beachten Sie, dass es sich hier um die Standarddatenbank **Default** im Ordner .../MapforceExamples handelt.
2. Klicken Sie auf die Auswahlliste "Globale Ressource" und wählen Sie den Eintrag "**releaseDB**".
3. Klicken Sie auf **Neu laden**, wenn die entsprechende Meldung angezeigt wird.
4. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Company xsi:schemaLocation="http://my-company.com/hamespace
  C:/DOCUME~1/MY/MYDOCU~1/Altova/MapForce2008/MapForceExamples
  http://www.w3.org/2001/XMLSchema" xmlns="http://my-company.com/h
  http://www.w3.org/2001/XMLSchema-instance">
3   <Person>
4     <First>Vernon</First>
5     <Last>Callaby</Last>
6     <Title>Office Manager</Title>
7     <PhoneExt>582</PhoneExt>
8     <Email>v.callaby@nanonull.com</Email>
9   </Person>
10  <Person>
11    <First>Frank</First>
12    <Last>Further</Last>
13    <Title>Accounts Receivable</Title>
14    <PhoneExt>471</PhoneExt>
15    <Email>f.further@nanonull.com</Email>
```

Die Daten aus der Release-Datenbank werden nun angezeigt. Da beide Datenbanken identisch sind, gibt es in diesem Ergebnis keinen sichtbaren Unterschied.

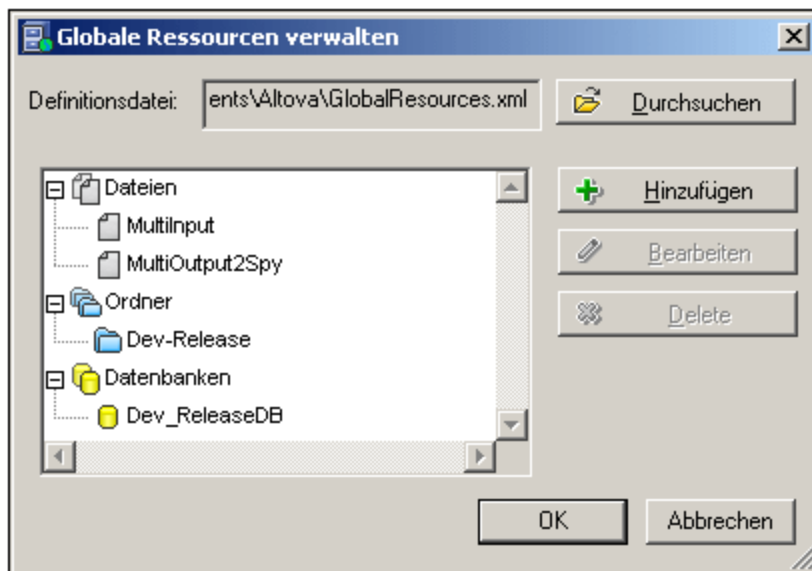
9.5 Globale Ressourcen - Eigenschaften

Die XML-Datei für globale Ressourcen

Die Definitionen der globalen Ressourcen sind in einer XML-Datei gespeichert. Standardmäßig hat diese XML-Datei den Namen `GlobalResources.xml` und ist im Ordner `C:\Documents and Settings\\My Documents\Altova\` gespeichert. Diese Datei ist für alle Altova-Applikationen als XML-Standarddatei für globale Ressourcen definiert. Aus diesem Grund steht eine globale Ressource, die in jeder beliebigen Altova-Applikation definiert wurde, allen Altova-Applikationen zur Verfügung - vorausgesetzt diese Datei wird in diesen Applikationen verwendet.

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, klicken Sie auf die **Durchsuchen**-Schaltfläche des Felds "Definitionsdatei" und wählen Sie die gewünschte im Dialogfeld "Öffnen" aus.



Verwalten globaler Ressourcen: Hinzufügen, Bearbeiten, Löschen

Im Dialogfeld "Globale Ressourcen" 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 drei Abschnitten aufgelistet: Dateien, Ordner und Datenbanken.

So fügen Sie eine globale Ressource hinzu:

Klicken Sie auf die Schaltfläche **Hinzufügen** und definieren Sie die globale Ressource im Dialogfeld "Globale Ressource", das angezeigt wird. Nachdem Sie eine globale Ressource definiert und gespeichert haben, wird die globale Ressource (d.h. der Alias) zur Liste der globalen Definitionen in der ausgewählten XML-Datei für globale Ressourcen hinzugefügt.

So bearbeiten Sie eine globale Ressource:

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.

So löschen Sie eine globale Ressource:

Um eine globale Ressource zu löschen, wählen Sie sie aus und klicken Sie auf **Löschen**.

So zeigen Sie das Ergebnis eines Applikationsarbeitsablaufs an:

Wenn die aufrufende Applikation, z.B. XMLSpy, eine andere Applikation, z.B. MapForce aufruft, so steht im Dialogfeld "Globale Ressourcen verwalten" eine Schaltfläche "Ansicht" zur Verfügung.

Wenn Sie auf die Schaltfläche "Ansicht" klicken, wird die Auswirkung der gerade ausgewählten globalen Ressource in der aufrufenden Applikation angezeigt. Ein Beispiel dazu finden Sie unter [Globale Ressourcen - Arbeitsablauf in Applikationen](#).

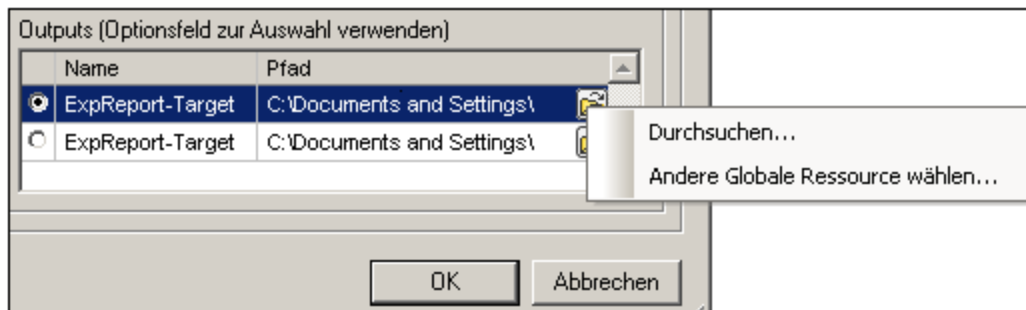
So speichern Sie Änderungen, die im Dialogfeld "Globale Ressourcen verwalten" vorgenommen wurden:

Nachdem Sie mit dem Hinzufügen, Bearbeiten oder Löschen globaler Ressourcen fertig sind, klicken Sie im Dialogfeld "Globale Ressourcen" auf **OK**, um Ihre Änderungen an der XML-Datei für globale Ressourcen zu speichern.

Anmerkung: Aliasressourcenamen müssen **innerhalb** von jedem der Abschnitte Dateien, Ordner und Datenbanken eindeutig sein. Sie können jedoch in zwei unterschiedlichen Abschnitten einen identischen Aliasnamen definieren. So kann es z.B. sowohl im Abschnitt Dateien als auch im Abschnitt Ordner einen Alias "multiInput" geben.

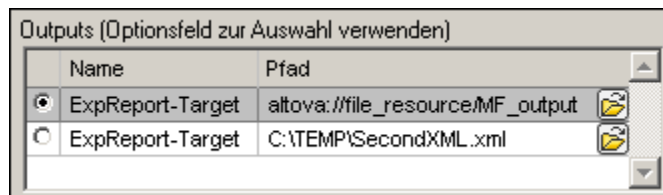
Auswahl von Ergebnissen vom MapForce-Transformationen als globale Ressource

In einer MapForce Transformation, die mehrere Output-Komponenten hat, können Sie auswählen, welche der Output-Dateien für die globale Ressource verwendet werden soll, indem Sie das Optionsfeld für diese Datei aktivieren.



Die durch das Mapping generierte **Ausgabedatei** kann gespeichert werden:

- über den Eintrag zur Auswahl einer anderen globalen Ressource im Popup Menü, der angezeigt wird als **altova://file_resource/MF_output**. Die Ausgabe wird in einer Datei gespeichert, die von der globalen Ressource referenziert wird.



- als Datei (über die Schaltfläche ) , angezeigt als C:\TEMP\Second.xml

Wenn keine dieser Optionen ausgewählt wird, wird eine **temporäre** XML-Datei erstellt, wenn die globale Ressource verwendet wird.

Festlegen, welche Ressource zur Laufzeit verwendet werden soll

Es gibt zwei applikationsweite Auswahloptionen, mit denen festgelegt wird, welche globalen Ressourcen verwendet werden und welche davon zu einem bestimmten Zeitpunkt tatsächlich verwendet wird:

- *die aktive XML-Datei für globale Ressourcen* wird im Dialogfeld "Globale Ressource" ausgewählt und kann jederzeit geändert werden. In diesem Fall werden die Definitionen der zuvor aktiven Datei sofort durch die der neuen aktiven Datei ersetzt.

Die aktive XML-Datei für globale Ressourcen legt also Folgendes fest: (i) welche globale Ressource zugewiesen werden kann 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 (bzw. der Dropdown-Liste in der Symbolleiste) wird eine Liste der Konfigurationen für alle Aliasse angezeigt.

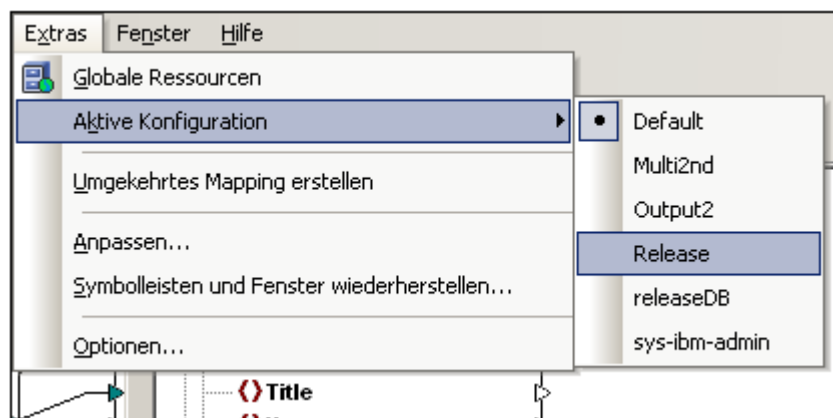
Bei Auswahl einer Konfiguration wird diese in der gesamten Applikation aktiv. D.h. überall, wo eine globale Ressource (oder ein Alias) verwendet wird, wird die Ressource, die der aktiven Konfiguration der einzelnen verwendeten Aliasse entspricht, geladen.

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

Wechseln von Ressourcen / Konfigurationen

Ressourcen können durch Auswahl eines anderen Konfigurationsnamen gewechselt werden. Dies kann auf zwei Arten erfolgen:

- Wenn Sie den Mauszeiger über den Menübefehl **Extras | Aktive Konfiguration** platzieren, wird ein Untermenü mit einer Liste aller Konfigurationen in der XML-Datei der globalen Ressourcen angezeigt. Wählen Sie die gewünschte Konfiguration aus dem Untermenü aus.



- Wählen Sie in der Auswahlliste der Symbolleiste "Globale Ressourcen" die gewünschte Konfiguration aus. Die Symbolleiste "Globale Ressourcen" kann über den Menübefehl

Extras | Anpassen, Klicken auf die Schaltfläche **Symbolleiste** und anschließendes Aktivieren/Deaktivieren des Kontrollkästchens "Globale Ressourcen" ein- bzw. ausgeblendet werden.



Kapitel 10

Dynamische Input/Output-Dateien pro Komponente

10 Dynamische Input/Output-Dateien pro Komponente

MapForce kann mehrere Input-/Output-Dateien pro Komponente verarbeiten und daher alle Dateien in einem Verzeichnis oder eine Teilmenge davon verarbeiten. Dies geschieht durch Verwendung von Platzhalterzeichen in der Input-Komponente.

Im Beispiel im [Tutorial](#) wird gezeigt, wie zwei XML-Input-Dateien von einer Quellkomponente verarbeitet werden und wie die Zielkomponente zwei XML-Dokumente ausgibt.

Für die folgenden Komponenten können mehrere Input- / Output-Dateien definiert werden:

- XML-Dateien
- Textdateien (CSV, Dateien mit fester Länge und FlexText-Dateien)
- EDI-Dokumente
- Excel Spreadsheets
- XBRL-Dokumente

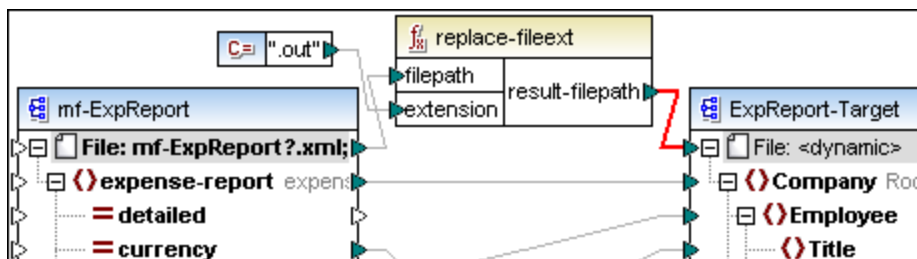
Beachten Sie bitte das Datenelement **Datei**: am oberen Rand der oben genannten Komponenten:



- Das Datenelement **Datei:mf-ExpReport.xml** von mf-ExpReport enthält als Eintrag die XML-Input-Datei. Dieser wird automatisch ausgefüllt, wenn Sie einer XML-Schema-Datei eine XML-Instanzdatei zuweisen. (Wenn eine Ausgabedatei definiert wurde, so wird auch der Name der Ausgabedatei angezeigt.)
- Das Datenelement **Datei: (Standard)** von ExpReport-Target zeigt an, dass der XML-Schema-Komponente keine Ausgabe-Instanzdatei zugewiesen wurde, als sie eingefügt wurde, d.h. die XML-Output-Datei ist leer. Daher wird bei der Ausführung des Mappings ein Standardwert verwendet.

Die Unterstützung dynamischer Dateinamen wird aktiviert, indem ein String, der einen Dateinamen enthält, auf das Datenelement File gemappt wird. Wenn die Komponente als Input-Komponente verwendet wird, kann der Dateiname Platzhalterzeichen enthalten. Siehe auch: [Behandlung relativer Pfade](#).

- Das Datenelement **Datei: <dynamic>** wird angezeigt, wenn eine Verbindung zum Datenelement "File" vorhanden ist, d.h. es werden nun mehrere Dateien unterstützt.
- Die Funktion **replace-fileext** konvertiert die Erweiterung .xml für die dynamischen Zieldateien in .out.



Dynamische Unterstützung/Unterstützung mehrerer Dateien und Unterstützung von

Platzhalterzeichen in von MapForce unterstützten Sprachen:

Zielsprache	Dynamischer Input-Dateiname	Unterstützung von Platzhaltern für Input-Dateinamen	Dynamischer Output-Dateiname
XSLT 1.0	*	wird von XSLT 1.0 nicht unterstützt	wird von XSLT 1.0 nicht unterstützt
XSLT 2.0	*	*(1)	*
XQuery	*	*(1)	wird von XQuery nicht unterstützt
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILTIN			

* unterstützt

- (1) Verwendet die Funktion **fn:collection**. Die Implementierung im **Altova** XSLT 2.0- und XQuery-Prozessor löst Platzhalterzeichen auf. Andere Prozessoren verhalten sich unter Umständen anders.

Die Platzhalterzeichen * und ? werden bei Eingabe ins Dialogfeld "Komponenteneinstellungen" und beim Mappen eines Strings auf den Node "File: Name" aufgelöst.

Informationen zum Transformieren von XSLT 1.0/2.0- und XQuery-Code mit AltovaXML finden Sie unter [Generieren von XSLT 1.0- und 2.0-Code](#) und [Generieren von XQuery 1.0-Code](#).

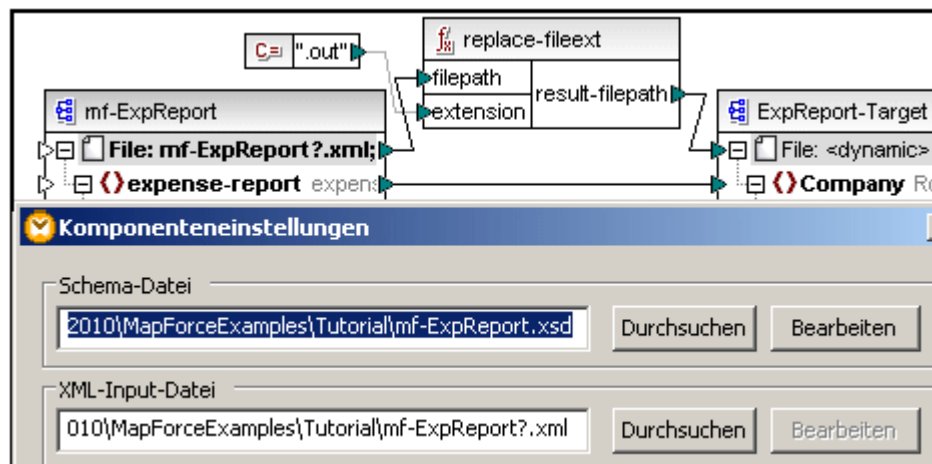
10.1 Dynamische Dateinamen - Input / Output

Durch dynamisches Mapping von Dateinamen innerhalb des Mappings können Sie:

- eine Mapping-Applikation generieren, in der die Input- und Output-Dateinamen während der Laufzeit generiert werden
- eine Gruppe von Dateien in ein anderes Format konvertieren (viele-in-viele)
- eine große Datei (oder Datenbank) in mehrere kleinere Abschnitte/Teile aufteilen
- mehrere Dateien in einer großen Datei zusammenführen (oder zu einer Datenbank zusammenführen)

Um mehrere Input-Dateien zu verarbeiten, stehen folgende Möglichkeiten zur Verfügung:

- Geben Sie im Dialogfeld **Komponenteneinstellungen** einen Dateipfad mit Platzhalterzeichen (* oder ?) als **Input-Datei** ein. Alle übereinstimmenden Dateien werden verarbeitet. Im Beispiel unten wird in der XML-Input-Datei das Platzhalterzeichen ? verwendet, um alle Dateien, die mit mf-ExpReport beginnen und ein weiteres Zeichen danach aufweisen, zu mappen. Im Ordner ...\\Tutorial gibt es davon zwei.



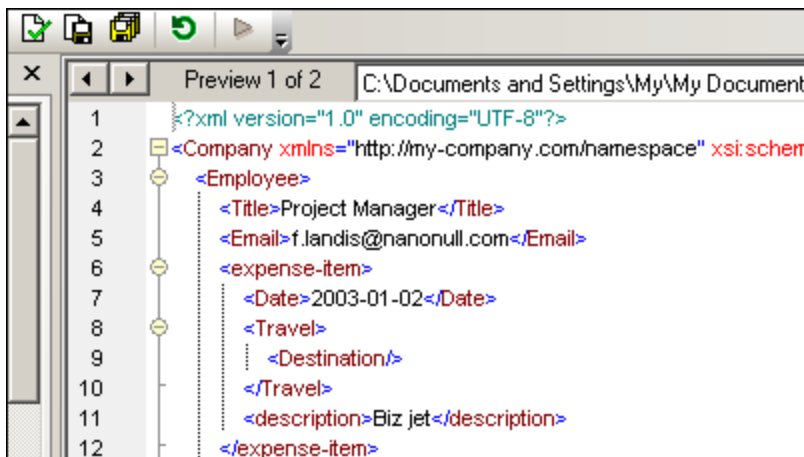
- Mappen Sie eine **Reihe** von Strings auf den *File*-Node der Quellkomponente. Jeder String in der Sequenz steht für einen Dateinamen. Die Strings können auch Platzhalterzeichen enthalten. Diese werden automatisch aufgelöst.

Eine Reihe von Dateinamen kann aus folgenden Quellen stammen:

- einer XML-Datei
- einer Textdatei (CSV oder feste Länge) usw.
- aus Datenbanktextfeldern
- aus einem Excel-Arbeitsblatt usw..

Vorschau auf dynamische Input- / Output-Mappings

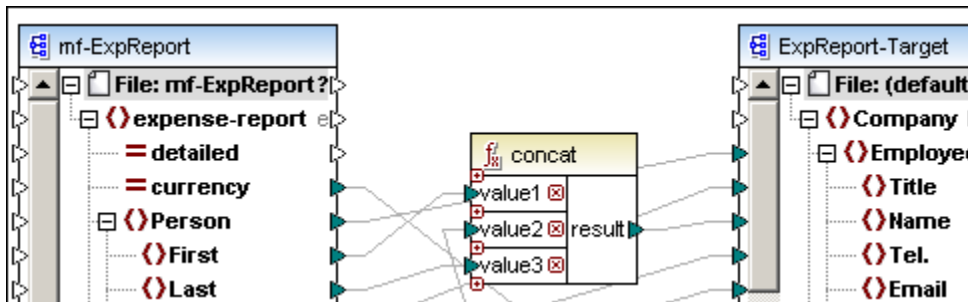
Wenn Sie auf die Schaltfläche "Ausgabe" klicken, werden die Mapping-Ergebnisse in einem Vorschaufenster angezeigt. Wenn beim Mapping mehrere Ausgabedateien erzeugt werden, wie in der Abbildung unten, Vorschau 1 von 2, gezeigt, hat jede Datei auf dem Ausgaberegister ihr eigenes nummeriertes Fenster. Klicken Sie auf die Pfeilschaltflächen, um die einzelnen Ausgabedateien zu sehen.



Klicken Sie auf die Schaltfläche "Alle generierten Ausgaben speichern" , um die hier gezeigten generierten Ausgabedateien zu speichern.

Mehrere Input-Dateien / eine Output-Datei - Zusammenführen von Input-Dateien

Mehrere Input-Dateien können in einer **einzigen** Output-Datei zusammengeführt werden, wenn der Konnektor **zwischen** den beiden **File**: Datenelementen entfernt wird, während in der Quellkomponente weiterhin z.B. über das Platzhalterzeichen "?" Zugriff auf mehrere Dateien besteht. Während also die Quellkomponente mehrere Dateien aufnehmen kann, kann dies die Output-Komponente nicht. Diese Dateien werden daher im Zieldokument aneinander angehängt.



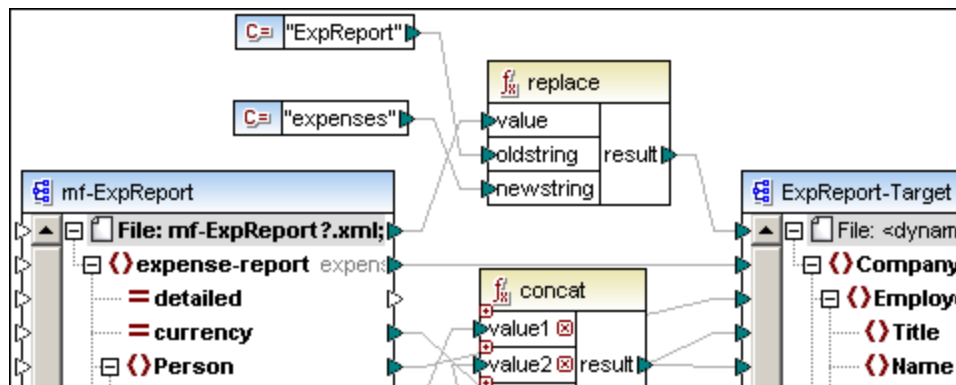
Mehrere Input-Dateien / mehrere Output-Dateien

Um mehrere Dateien n:n auf mehrere Zieldateien zu mappen, müssen Sie eindeutige Output-Dateinamen generieren. In einigen Fällen können die Output-Dateinamen von Strings in den Input-Daten abgeleitet werden, in anderen Fällen ist es sinnvoll, den Output-Dateinamen vom Input-Dateinamen abzuleiten, z.B. durch Änderung der Dateierweiterung.

Der vollständige Pfadname der aktuell verarbeiteten Datei steht zur Verfügung, indem Sie das Output-Symbol des *File* Node z.B. mit "concat" verbinden, um eine neue Dateierweiterung hinzuzufügen.

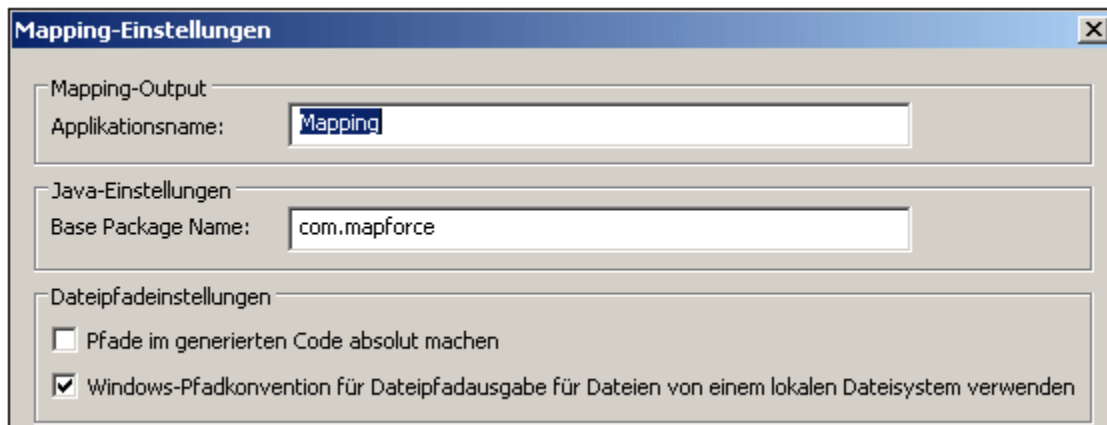
Bitte beachten Sie:

Vermeiden Sie es, einfach die *File*: Nodes **direkt** und ohne Verwendung von Verarbeitungsfunktionen miteinander zu verbinden, da sonst Ihre Input-Dateien bei Ausführung des Mappings überschrieben werden. Sie können die Output-Dateinamen mit Hilfe einer Reihe von Funktionen wie z.B. der unten gezeigten *replace*-Funktion, ändern.



Im obigen Fall lauten die Output-Dateinamen **mf-expenses1.xml** und **mf-expenses2.xml**.

Mit Hilfe der Menüoption **Datei | Mapping-Einstellungen** können Sie die für das Mapping-Projekt verwendeten Dateipfadeinstellungen global definieren.



Über das Kontrollkästchen "Windows-Pfadkonvention...verwenden" können Sie sicherstellen, dass die Windows-Pfadkonventionen verwendet werden. Bei Ausgabe von XSLT2 (und XQuery) wird der Name der aktuell verarbeiteten Datei intern mit Hilfe der **document-uri** Funktion, die für lokale Dateien einen Pfad in der Form **file:// URI** zurückgibt, abgerufen.

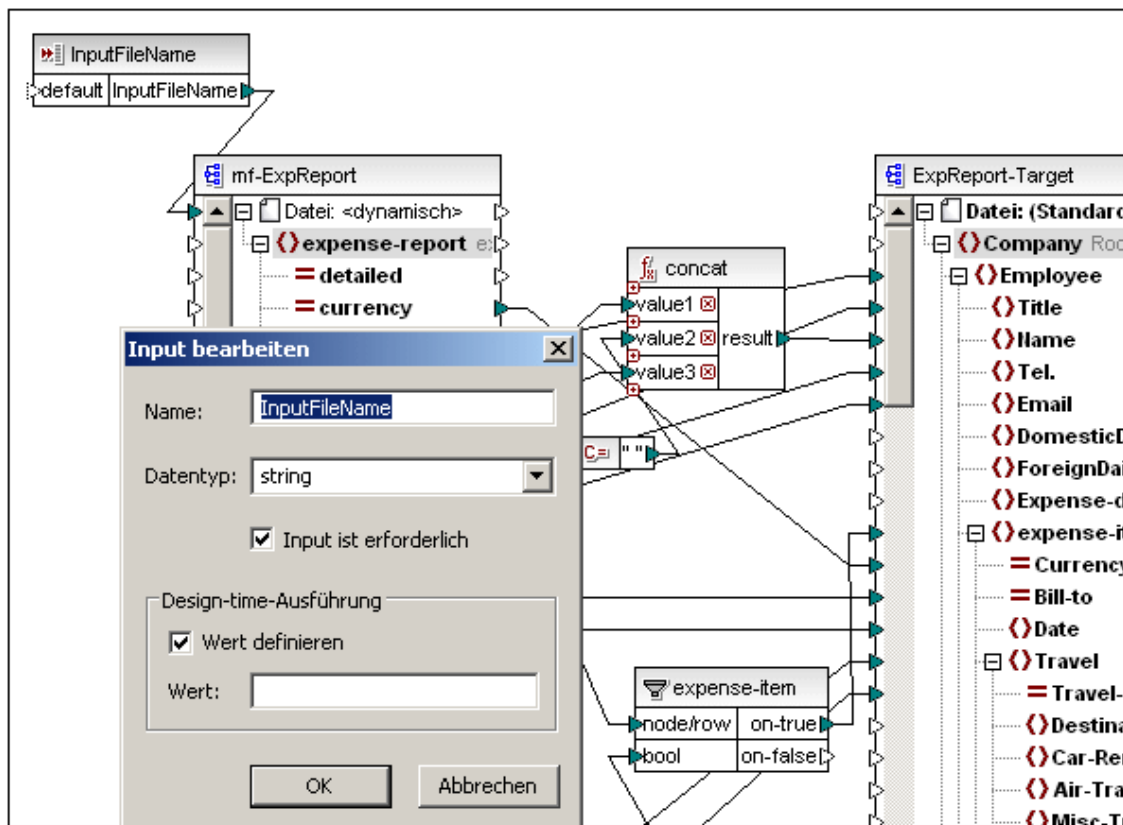
Wenn dieses Kontrollkästchen aktiv ist, wird eine file:// URI-Pfadspezifikation automatisch in einen vollständigen Windows-Dateipfad (z.B. "C:\...") konvertiert, um die Weiterverarbeitung zu vereinfachen.

10.2 Dynamische Dateinamen als Input-Parameter

Sie können in MapForce spezielle Input-Komponenten erstellen, die bei Ausführung des kompilierten Mappings über die Befehlszeile als **Parameter** verwendet werden können. Dieser spezifischen Input-Komponententyp kann nicht **innerhalb** einer benutzerdefinierten Funktion verwendet werden, sondern steht nur im Haupt-Mapping-Fenster zur Verfügung.

So verarbeiten Sie eine Datei mit Hilfe eines Input-Parameters während der Laufzeit:

Um den Pfad und den Dateinamen während der Laufzeit zu definieren, verbinden Sie eine [Input-Parameter](#)-Komponente mit dem Input-Symbol des *File* Node in der Quellkomponente. Je nachdem, welche Verbindungen zu anderen Datenelementen bestehen, wird dadurch der Input- und/oder Output-Dateiname definiert.



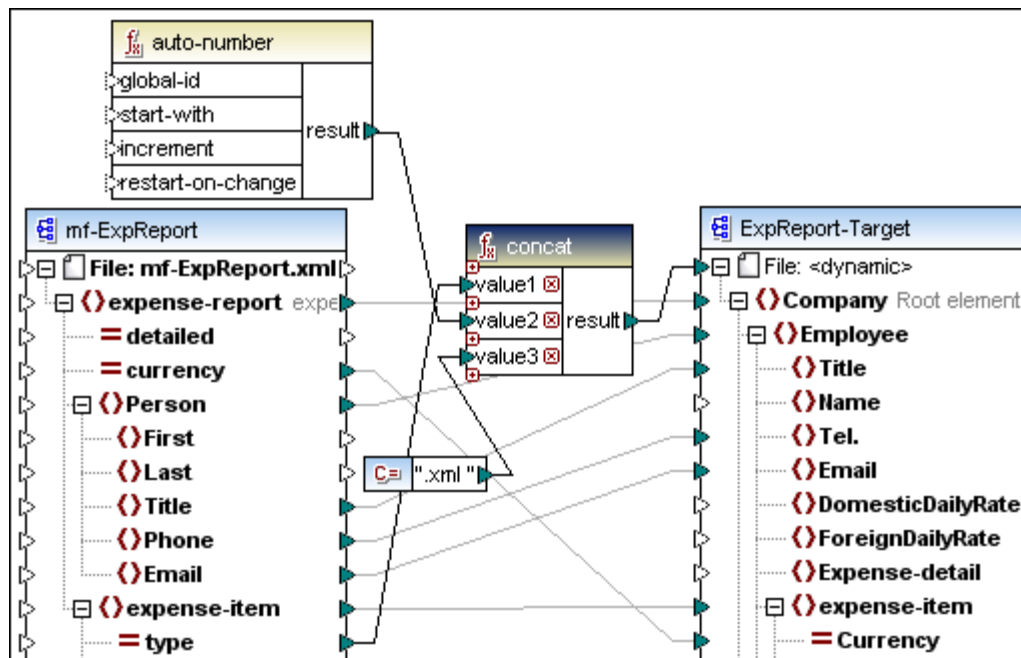
Der Eintrag mf-ExpReport.xml im Feld "Wert" wird nur zu Vorschauzwecken im Ausgabefenster verwendet. Er hat keine Auswirkung auf die Parameterwerte, die bei Ausführung des Codes über die Befehlszeile verwendet werden. Nähere Informationen dazu finden Sie unter [Befehlszeile - Definieren von Input/Output-Dateien](#).

Nach Generierung und Kompilierung des Codes können Sie den Dateinamen für das Mapping über die Befehlszeile bereitstellen:

mapping.exe /Input-Dateiname Dateiname.xml.

wobei:

- **/Input-Dateiname** der Name des ersten Parameters ist
- **Dateiname.xml** der zweite Parameter ist, d.h. der dynamische Dateiname, der verwendet werden soll, wenn die Applikation über die Befehlszeile ausgeführt wird.



4. Erstellen Sie die Verbindungen wie oben gezeigt: type zu value1, auto-number zu value2 und die Konstante zu value3.
5. Verbinden Sie den **result**-Parameter/Output der concat-Funktion mit dem Datenelement **File:** der Zielkomponente. Beachten Sie, dass nun File: <dynamic> angezeigt wird.
6. Definieren Sie die restlichen benötigten Verbindungen.
7. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

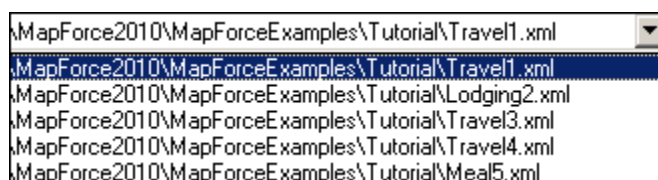
```

Vorschau 1 von 5 C:\Documents and Settings\anf\My Documents\Altova\MapForce2010\MapForceExamples\Tutorial\Travel1.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Company xmlns="http://my-company.com/namespace" xsi:schemaLocation="http://my-company.com/namespace ExpReport-Target
  ="http://www.w3.org/2001/XMLSchema-instance">
3   <Employee>
4     <Title>Project Manager</Title>
5     <Tel.>123-456-78</Tel.>
6     <Email>f.jandis@nanonull.com</Email>
7     <expense-item Currency="USD" Bill-to="Development">
8       <Date>2003-01-02</Date>
9       <Travel Travel-Cost="337.88">
10        <Destination/>
11      </Travel>
12      <description>Biz jet</description>
13    </expense-item>
14  </Employee>
15 </Company>
16

```

Jeder Datensatz wird nun auf einem eigenen Vorschauregister angezeigt. Oben sehen Sie den ersten davon.

8. Klicken Sie auf den Pfeil der Dropdown-Liste, um alle generierten Dateien zu sehen.



Wenn Sie auf die Pfeile "Vorheriges" und "Nächstes"  klicken, sehen Sie die einzelnen Dateien auf dem Ausgaberegister.

Anmerkung:

- Das Attribut **type** liefert den ersten Teil des Dateinamens, z.B. Travel.
- Die Funktion **auto-number** liefert die fortlaufend nummerierten Dateinummern (die Standardeinstellung ist Beginn bei=1 und erhöhen um=1) - daher Travel1.
- Die **Konstantenkomponente** liefert die Dateierweiterung, d.h. .xml, also lautet der Dateiname der ersten Datei Travel1.xml.
- Durch Klicken auf die Schaltfläche "Alle speichern"  können Sie die einzelnen Dateien direkt über das Ausgaberegister speichern, ohne Code generieren zu müssen.

10.4 Mehrere XML-Dateien pro Tabelle

Unten sehen Sie den Inhalt der Datenbankdatei **altova.mdb** aus dem Ordner ...\\MapForceTutorial. Die Datei besteht aus vier Tabellen, wobei die Tabelle "Person" 21 separate Person-Datensätze enthält. Dieses Beispiel steht unter dem Namen **PersonDB-dyn.mfd** im Ordner ...\\Tutorial zur Verfügung.

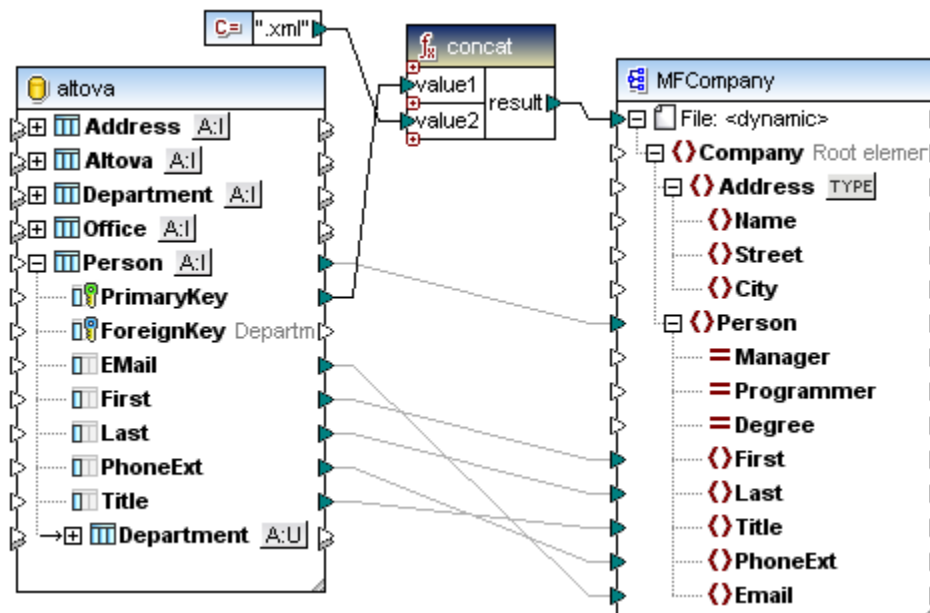
Ziel:

Die Generierung einer separaten XML-Datei für die einzelnen unten angezeigten Personendatensätze (Person).

Person					
PrimaryKey	ForeignKey	E-Mail	First	Last	
1	1	v.callaby@nanonu	Vernon	Callaby	
2	1	f.further@nanonu	Frank	Further	
3	1	l.matisse@nanonu	Loby	Matisse	
4	2	j.firstbread@nanc	Joe	Firstbread	
5	2	s.sanna@nanonu	Susi	Sanna	
6	3	f.landis@nanonu	Fred	Landis	
7	3	m.landis@nanonu	Michelle	Butler	
8	3	t.little@nanonull	Ted	Little	

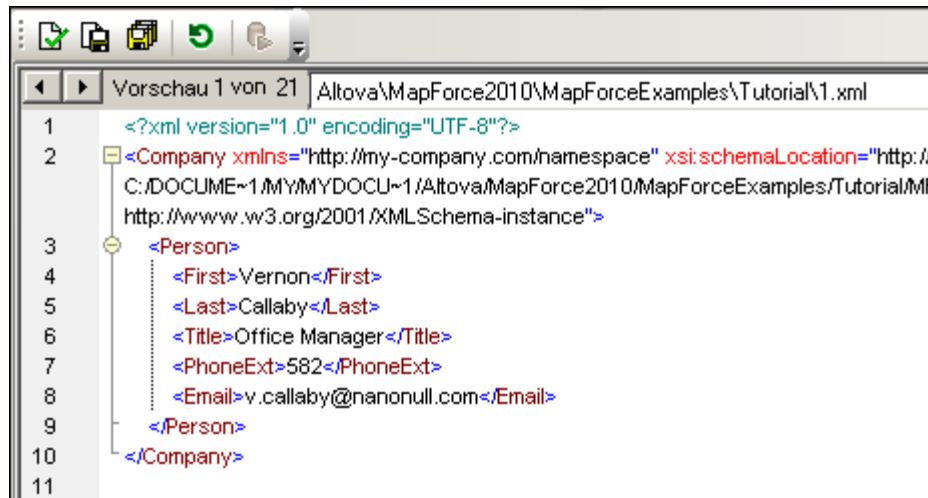
Da das Feld "PrimaryKey" die einzelnen Personen in der Tabelle definiert, ist dies das Datenelement, das wir zum Aufteilen der Quelldatenbank in separate Dateien verwenden werden.

1. Fügen Sie eine concat-Funktion aus dem Bibliotheksfenster und über die Symbolleiste eine Konstantenkomponente ein.
2. Geben Sie als String-Wert der Konstantenkomponente *.xml ein, wenn das Dialogfeld angezeigt wird.



3. Erstellen Sie die Verbindungen wie oben gezeigt: PrimaryKey zu value1 und die

- Konstante zu value2.
4. Verbinden Sie den **result**-Parameter/Output der concat-Funktion mit dem Datenelement **File:** der Zielkomponente. Beachten Sie, dass nun File: <dynamic> angezeigt wird.
 5. Definieren Sie die restlichen benötigten Verbindungen.
 6. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Jeder Datensatz wird nun auf einem eigenen Vorschauregister angezeigt. Oben sehen Sie den ersten davon.

Wenn Sie auf die Pfeile "Vorheriges" und "Nächstes"  klicken, sehen Sie die einzelnen Dateien auf dem Ausgaberegister.

Anmerkung:

- Das Feld **PrimaryKey** liefert den ersten Teil des Dateinamens, z.B. 1.
- Die **Konstantenkomponente** liefert die Dateierweiterung, d.h. **.xml**, also lautet der Dateiname der ersten Datei 1.xml.
- Durch Klicken auf die Schaltfläche "Alle speichern"  können Sie die einzelnen Dateien direkt über das Ausgaberegister speichern, ohne Code generieren zu müssen.

10.5 Mehrere XML-Dateien anhand von Excel-Zeilen

Unten sehen Sie den Inhalt der Spreadsheet-Datei **altova.xlsx** aus dem Ordner ...\\MapForceExamples\\Tutorial. Die Datei besteht aus zwei Arbeitsblättern: Admin mit 10 und Development mit 11 Datenzeilen. Dieses Beispiel steht unter dem Namen

Excel-Mapping-dyn.mfd im Ordner ...\\Tutorial zur Verfügung.

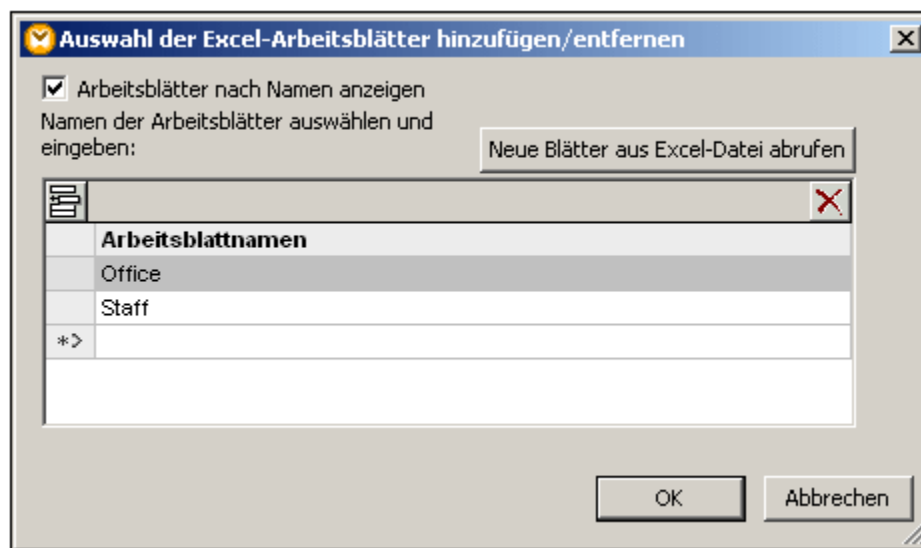
Arbeitsblatt "Admin"

	A	B	C	D
1	Valentin	Bass	716	v.bass@nanon
2	Theo	Bone	331	t.bone@nanon
3	Vernon	Callaby	582	v.callaby@nanon
4	Joe	Firstbread	621	j.firstbread@nanon
5	Frank	Further	471	f.further@nanon
6	Alex	Martin	778	a.martin@nanon
7	Loby	Matise	963	l.matise@nanon
8	Steve	Meier	114	s.meier@nanon
9	Max	Nafta	122	m.nafta@nanon
10	Susi	Sanna	753	s.sanna@nanon
11				
12				

Arbeitsblatt "Development"

	A	B	C	D
1	Jessica	Bander	241	j.band@na
2	Michelle	Butler	654	m.landis@
3	Carl	Franken	147	c.franken@
4	Liz	Gardner	753	l.gardner@
5	George	Hammer	223	g.hammer@
6	Lui	King	345	l.king@na
7	Fred	Landis	951	f.landis@n
8	Ted	Little	852	t.little@na
9	Mark	Redgreen	152	m.redgreer
10	Paul	Smith	334	p.smith@n
11	Ann	Way	951	a.way@na
12				

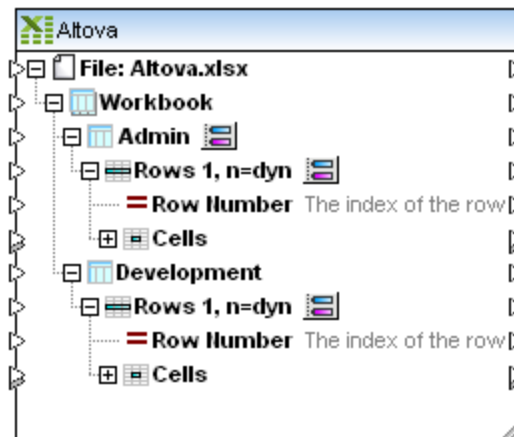
MapForce kann Excel-Komponenten je nach Komponentenooptionen auf zwei verschiedene Arten anzeigen und mappen. Die Standardeinstellung wird im Dialogfeld unten angezeigt. Das Kontrollkästchen "Arbeitsblätter nach Namen anzeigen" ist aktiv, wenn Sie die Excel-Komponente zum ersten Mal einfügen.



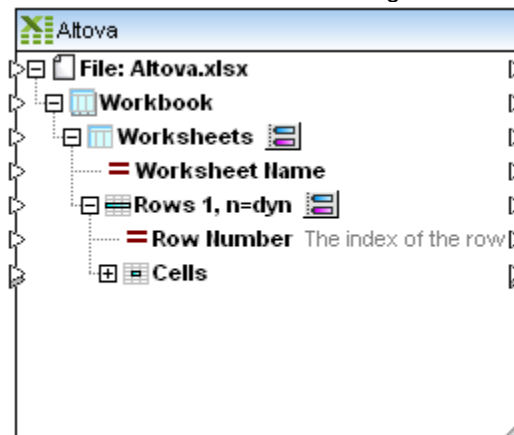
So rufen Sie eine Arbeitsmappe auf, als würde es sich um ein einziges Arbeitsblatt

handeln:

1. Klicken Sie in der Excel-Komponente auf das Symbol  neben **Admin**.



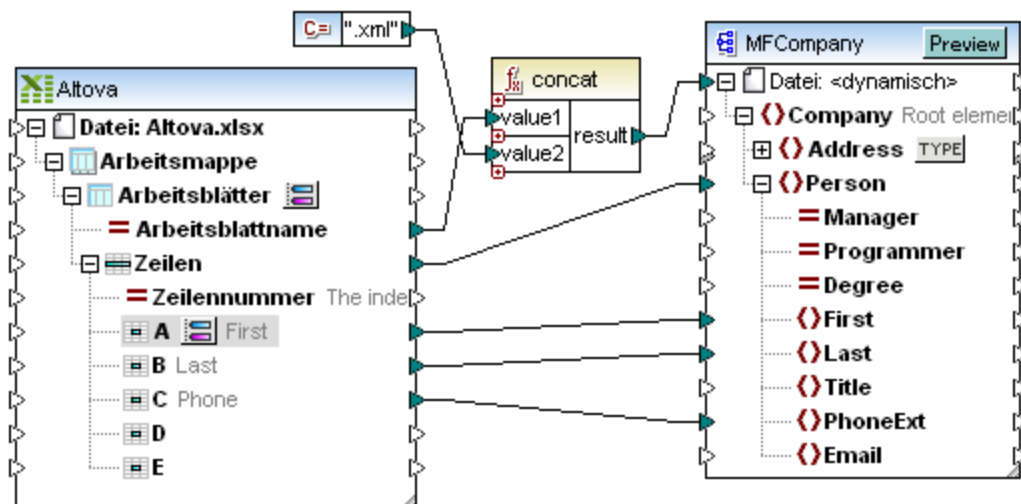
2. Klicken Sie auf das Kontrollkästchen "Arbeitsblätter nach Namen anzeigen", um es zu deaktivieren.
Die benannten Arbeitsblätter sind nun nicht mehr zu sehen (siehe Abbildung unten).
Statt dessen steht nun ein Eintrag "Arbeitsblattname" zur Verfügung.



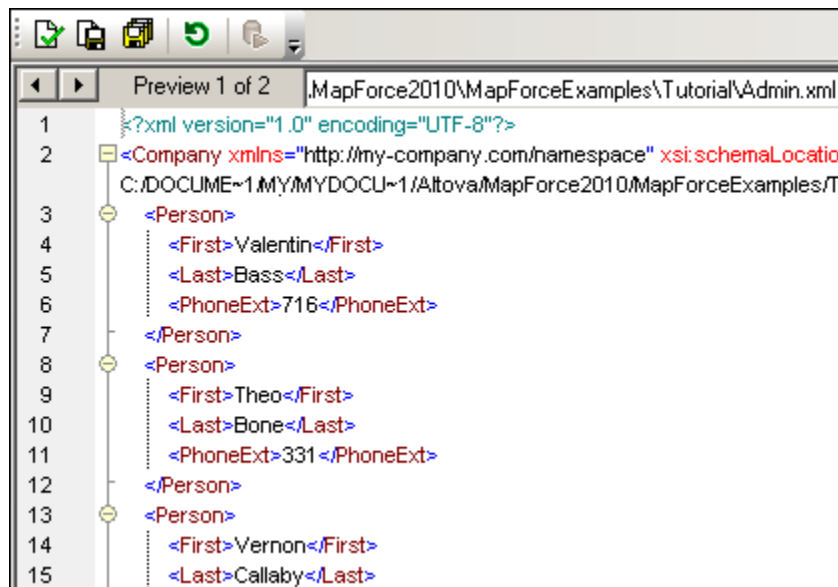
Ziel 1: Die Generierung separater Dateien für die einzelnen Arbeitsblätter, die die Personendatensätze jedes Arbeitsblatts enthalten.

Das Datenelement "Arbeitsblattname" legt die spezifischen Arbeitsblätter in der Arbeitsmappe fest, daher wird dieses Datenelement zum Aufteilen der Quelle in separate Dateien verwendet.

1. Fügen Sie aus dem Bibliotheksfenster eine concat-Funktion und eine Konstantenfunktion ein.



- Erstellen Sie die Verbindungen wie oben gezeigt: Arbeitsblattname zu value1 und die Konstante zu value2.
- Verbinden Sie den **result**-Parameter der concat-Funktion mit dem Datenelement **File** der Zielkomponente. Beachten Sie, dass nun File: <dynamic> angezeigt wird.
- Definieren Sie die restlichen benötigten Verbindungen.
- Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Jeder Datensatz wird nun auf einem eigenen Vorschauregister angezeigt. Oben sehen Sie den ersten davon.

Wenn Sie auf die Pfeile "Vorheriges" und "Nächstes"  klicken, sehen Sie die einzelnen Dateien auf dem Ausgaberegister.

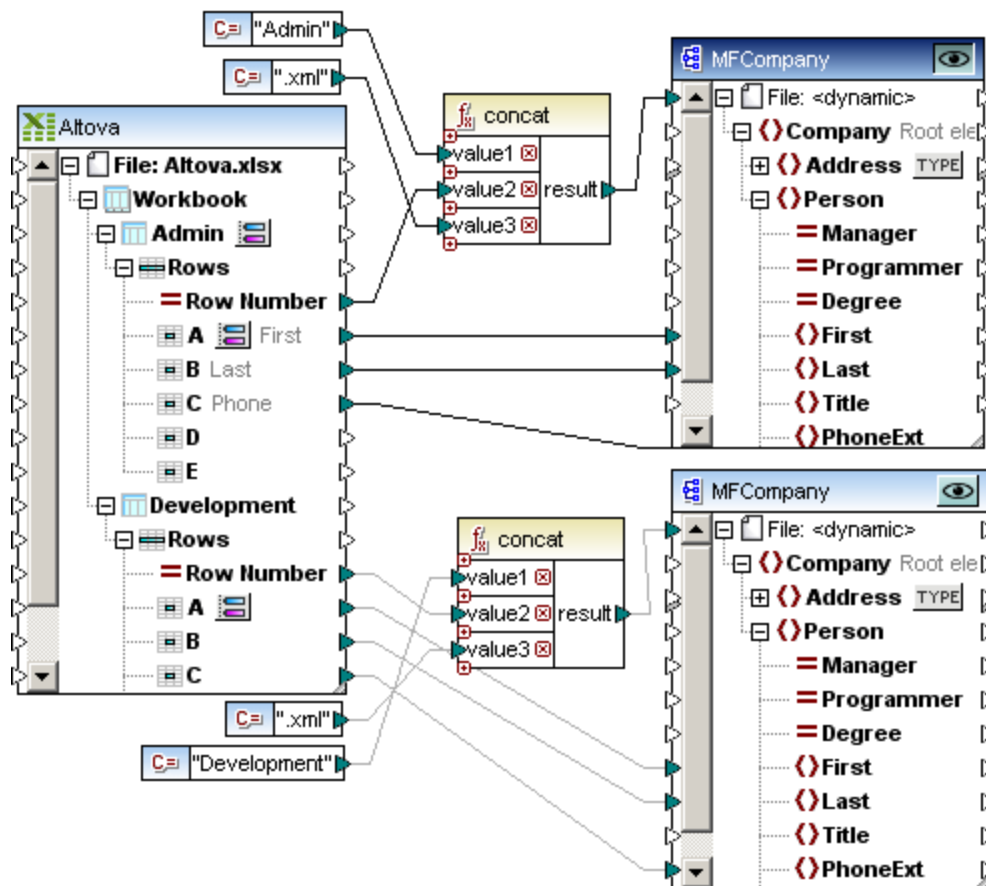
Anmerkung:

- Das Feld **Arbeitsblatt** liefert den ersten Teil des Dateinamens, also Admin.
- Die **Konstantenkomponente** liefert die Dateierweiterung, d.h. **.xml**, also lautet der Name der ersten Datei Admin.xml. Admin.xml enthält alle Zeilen dieses Excel-Arbeitsblatts. Development.xml enthält die anderen Zeilen.

- Durch Klicken auf die Schaltfläche "Alle speichern"  können Sie die einzelnen Dateien direkt über das Ausgaberegister speichern, ohne Code generieren zu müssen.

Ziel 2: Die Generierung separater Dateien für jede einzelne Person in jedem Arbeitsblatt

1. Klicken Sie auf die Schaltfläche  neben den Arbeitsblättern, wenn Sie den Abschnitt oben erstellt haben und aktivieren Sie das Kontrollkästchen "Arbeitsblätter nach Namen anzeigen".
In der Komponenten werden nun die einzelnen Arbeitsblätter angezeigt: Admin und Development.



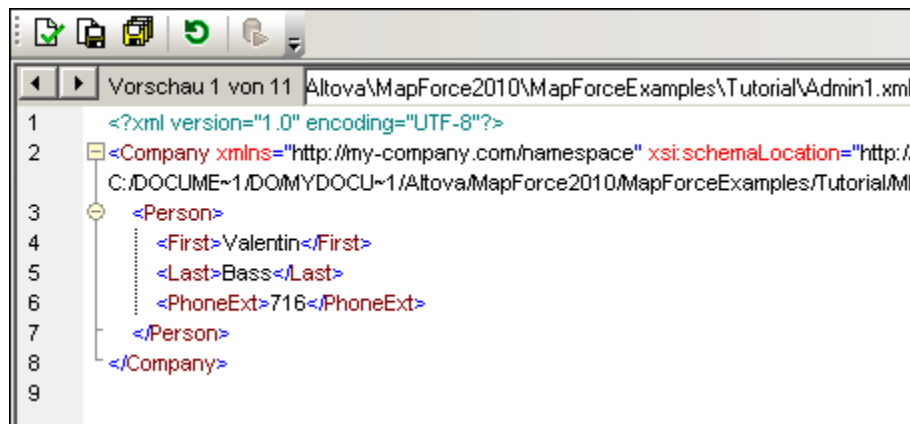
2. Fügen Sie die concat-Funktionen und die Konstantenfunktionen wie gezeigt ein.
3. Fügen Sie eine zweite XML-Schema-Datei mit demselben Namen ein und erstellen Sie die Verbindungen wie gezeigt.

Da das Element "Zeilennummer" die einzelnen Personenzeilen im Arbeitsblatt bestimmt, ist dies das Datenelement, das wir zum Aufteilen der einzelnen Arbeitsblätter in separate Dateien verwenden werden.

Gehen Sie nun beim **Admin**-Arbeitsblatt-Datenelement in der Altova XLSX-Komponente folgendermaßen vor:

1. Erstellen Sie die Verbindungen wie oben gezeigt: Admin-Konstante zu value1, **Zeilennummer** zu value2 und .xml-Konstante zu value3.
2. Verbinden Sie den **result**-Parameter der concat-Funktion mit dem Datenelement **File**:

- der Zielkomponente. Beachten Sie, dass nun File: <dynamic> angezeigt wird.
4. Definieren Sie die restlichen benötigten Verbindungen.
 5. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Jeder Datensatz wird nun auf einem eigenen Vorschauregister angezeigt. Alle 10 Datensätze des Admin-Arbeitsblatts wurden auf separate Dateien aufgeteilt.

Wenn Sie auf die Pfeile "Vorheriges" und "Nächstes"  klicken, sehen Sie die einzelnen Dateien auf dem Ausgaberegister.

Anmerkung:

- Das Feld **Arbeitsblatt** liefert den ersten Teil des Dateinamens, z.B. Admin.
- Die **Zeilennummer** liefert die Zeilennummer aus dem Excel-Arbeitsblatt, z.B. 1.
- Die **Konstantenkomponente** liefert die beim Speichern der Datei verwendete Dateierweiterung, also Admin1.xml, wie oben gezeigt.
- Durch Klicken auf die Schaltfläche "Alle speichern"  können Sie die einzelnen Dateien direkt über das Ausgaberegister speichern, ohne Code generieren zu müssen.

Um die Ausgabe des Arbeitsblatts "Development" zu sehen, klicken Sie auf die Schaltfläche "Preview" dieser Komponente und anschließend auf das Ausgaberegister, um das Ergebnis des Mapping zu sehen. 11 Datensätze wurden auf separate Dateien aufgeteilt.

10.6 Relative und absolute Dateipfade

Ein relativer Pfad ist ein Pfad, der nicht mit einem Laufwerksbuchstaben beginnt, d.h. es kann sich um einen Dateinamen ohne einen Pfad handeln. Der spezifische Kontext, in dem der relative Dateiname verwendet wird, definiert den Basispfad. Die Behandlungsweise von relativen Dateinamen hat sich aufgrund der Unterstützung für das Mappen von Dateinamen als Daten in einem Mapping in MapForce Version 2010 geändert.

In früheren Versionen von MapForce (vor 2010) wurden Dateipfade für Dateien im selben oder einem untergeordneten Ordner relativ zur *.MFD-Datei gespeichert und beim Öffnen/Laden in absolute Pfade geändert.

Seit MapForce 2010 werden alle Referenzen auf externe Dateien wie z.B. Schemas oder XML-Instanzdateien so gespeichert, wie sie ins Dialogfeld eingegeben wurden. Auf diese Art können relative Pfade verwendet werden, wo dies erforderlich ist.

Alle Pfade relativ zur MFD-Datei speichern

Diese neue Option, die in allen Komponenteneinstellungsdialogfeldern zur Verfügung steht, speichert alle Dateipfade (der Komponente) relativ zum Ordner der aktuellen MFD-Datei. Auf diese Art können Sie ein Mapping zusammen mit allen damit in Zusammenhang stehenden Dateien in einen anderen Ordner verschieben, wobei alle Dateireferenzen weiterhin gültig bleiben.

Das bedeutet dass:

- absolute Dateipfade in relative Pfade geändert werden
- auch das übergeordnete Verzeichnis "..\\" eingefügt/geschrieben wird
- die Dateipfade bei Verwendung von "Speichern unter..." (relativ zur MFD-Datei) an den neuen Ordner, in dem die MFD-Datei gespeichert wird, angepasst werden.

Bitte beachten Sie:

Pfade, die ein nicht lokales Laufwerk referenzieren oder bei denen eine URL verwendet wird, werden nicht zu relativen Pfaden gemacht.

Komponenteneinstellungen

Komponentenname:

Schema-Datei:

XML-Input-Datei:

XML-Output-Datei:

Target-Namespace-Präfix:

☒ Schema- / DTD-Referenz hinzufügen (leeres Feld für absoluten Pfad des Schemas):

☒ Werte in Zieltypen konvertieren (deaktivieren, um numerische oder Datumswerte beizubehalten, auf die Gefahr hin, dass eine ungültige Ausgabe erzeugt wird)

☒ Pretty Print für Ausgabe

☐ Digitale Signatur erstellen (nur Built-in-Ausführung)

Falls Signatur nicht erstellt werden kann: ☐ Verarbeitung stoppen ☒ Ohne Signatur fortfahren

Kodierung

Kodierungsname:

Bytefolge: ☐ Bytefolge-Markierung inkl.

StyleVision Power Stylesheet-Datei:

☒ Input-Verarbeitungsoptimierungen auf Basis von min/maxOccurs aktivieren

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Es gibt zwei separate Arten von Dateien, die von einer MFD-Datei aus referenziert werden:

- Dateien vom Typ Schema (XML-Schemas, WSDL-, FlexText-Konfigurationsdateien, ...), die in das Feld **Schema-Datei** eingegeben werden.
- Instanzdateien, die in das Feld **xxx-Input-Datei** oder **xxx-Output-Datei** eingegeben werden.

Dateien vom Typ Schema

Dateien vom Typ Schema werden beim Erstellen eines Mappings verwendet. Sie definieren die **Struktur** der gemappten Input- und Output-Instanzdateien. Anhand dieser Informationen wird die Baumstruktur/Hierarchie der Datenelemente der verschiedenen Komponenten angezeigt. MapForce unterstützt das Eingeben und Speichern eines relativen Pfads zu Dateien vom Typ Schema.

- Relative Pfade zu Dateien **vom Typ Schema** werden immer **relativ zur MFD-Datei** aufgelöst.
- Wenn Sie ein Schema über das Dialogfeld "Öffnen", z.B. nachdem eine neue Komponente eingefügt wurde, oder durch Klicken auf die Schaltfläche "Durchsuchen" im Dialogfeld "Komponenteneinstellungen" auswählen, wird immer der **absolute** Pfad in das Feld eingefügt.
- Um einen relativen Pfad zu definieren, der ebenfalls in der MFD-Datei gespeichert wird, löschen Sie den Pfad aus dem Textfeld oder geben Sie einen relativen Pfad oder Dateinamen ein. Dies geschieht automatisch beim Speichern der MFD-Datei, wenn das Kontrollkästchen "Alle Pfade relativ zur MFD-Datei speichern" aktiviert ist. Sie können den übergeordneten Ordner der MFD-Datei auch mit "..\\" definieren.
- Wenn Sie eine MFD-Datei, die Dateien aus demselben Verzeichnis referenziert, speichern und das ganze Verzeichnis anschließend in einen anderen Ordner verschieben, werden **absolute** in der mfd-Datei gespeicherte Pfade **nicht aktualisiert**. Daher sollten Benutzer, die Versionskontrollsysteme und unterschiedliche Arbeitsverzeichnisse verwenden, in diesem Fall relative Pfade verwenden.

Instanzdateien und die Ausführungsumgebung

Die Verarbeitung von Instanzdateien erfolgt in der generierten XSLT oder XQuery-Datei oder in der generierten Applikation sowie in der MapForce-Vorschau.

In den meisten Fällen ist es nicht sinnvoll, relative Pfade zu Instanzdateien als relativ zur MFD-Datei zu interpretieren, da dieser Pfad zum Zeitpunkt der Ausführung möglicherweise gar nicht existiert - der generierte Code könnte ja auf einem anderen Rechner ausgeführt werden.

Relative Dateinamen für Instanzdateien werden daher relativ zur **Ausführungsumgebung** aufgelöst.

Zielsprache	Basispfad für relativen Instanzdateinamen
XSLT/XSLT2	Pfad der XSLT-Datei
XQuery	Pfad der XQuery-Datei
C++, C#, Java	Arbeitsverzeichnis bei der Ausführung des generierten Codes
MapForce-Vorschau/BUILTIN (oder Ausführung über API)	Pfad der MFD-Datei
BUILTIN-Ausführung über die Befehlszeile	In der Befehlszeile definiert (Zielverzeichnis)

Zum Dialogfeld **Datei | Mapping-Einstellungen** wurde ein neues Kontrollkästchen hinzugefügt, **"Pfade in generiertem Code absolut machen"**, das die Kompatibilität von generiertem Code mit Mapping-Dateien (*.mfd) aus Versionen vor Version 2010 gewährleistet.

Der Status des Kontrollkästchens wird automatisch eingestellt und hängt davon ab, welche

Dateien geöffnet sind. Das Kontrollkästchen ist:

- **deaktiviert**, wenn eine **neue** Mapping-Datei, d.h. Version 2010 erstellt oder geöffnet wird
Relative Pfade für Input- und Output-Instanzdateien werden im aktuellen Zustand in den generierten Code geschrieben.
Dies ermöglicht die Verwendung des generierten Codes in einem anderen Verzeichnis oder sogar auf einem anderen Rechner. Sie müssen sicherstellen, dass die mit relativen Pfaden angegebenen Dateien in der Laufzeitumgebung im richtigen Ordner zur Verfügung stehen.
- **aktiviert**, wenn eine ältere Mapping-Datei aus Version 2009, 2008 usw. geöffnet ist
Relative Pfade für Input- und Output-Instanzdateien werden vor der Generierung von Code absolut gemacht (relativ zur *.MFD-Datei). Dies hat denselben Effekt, wie die Generierung von Code mit einer älteren Version von MapForce.

Beachten Sie, dass der **Quellinstanzdateiname** auch für folgende Zwecke verwendet wird:

- Ermittlung des XML-Root-Elements und des referenzierten Schemas
- Validierung anhand des ausgewählten Schemas
- Lesen von Excel Arbeitsblattnamen und -spalten
- zum Lesen von Spaltennamen und der Vorschau auf den Inhalt von Textdateien (CSV oder FLF)

Neues "schemaLocation" Feld für XML-Zieldateien

Da Schemareferenzen relativ zur MFD-Datei gespeichert werden können und sich die anhand einer Zielkomponente **generierte** XML-Datei oft in einem anderen Verzeichnis befindet, gibt es eine Methode, einen separaten **schemaLocation**-Pfad für die **XML-Zielinstanz** einzugeben, damit die XML-Datei dort validiert werden kann.

Dies ist das Feld unterhalb des Kontrollkästchens "Schema/DTD-Referenz hinzufügen" im Dialogfeld "Komponenteneinstellungen" (Doppelklicken Sie auf eine Komponente, um es zu öffnen). Ein ähnliches Feld gibt es für die Taxonomiereferenz in XBRL-Komponenten.

Der Pfad des referenzierten/verknüpften Schemas, der in dieses Feld eingegeben wird, wird in den generierten **Zielinstanzdateien** ins Attribut **xsi:schemaLocation** oder in die DOCTYPE-Deklaration geschrieben, wenn eine DTD verwendet wird.

Bei einer XBRL-Zieldatei wird die Taxonomiereferenz in das href-Attribut des Elements link:schemaRef geschrieben.

Anmerkung: Auch eine URL wie z.B. <http://mylocation.com/mf-expreport.xsd> kann hier eingegeben werden.

Kapitel 11

Zwischenvariablen

11 Zwischenvariablen

Bei Zwischenvariablen handelt es sich um eine spezielle Art von Komponente, die zur Lösung verschiedener [komplexer Mapping-Probleme](#) verwendet wird. Zwischenvariablen dienen zum Speichern von Mapping-Zwischenergebnissen für die weitere Verarbeitung.

- Variablen funktionieren in allen Sprachen mit Ausnahme von XSLT1.0.
- Variablenergebnisse sind immer Sequenzen, also eine Liste getrennter Werte, und können auch zum Erstellen von Sequenzen verwendet werden.
- Variablen sind Strukturkomponenten mit einem Root Node und haben keine Instanzen (XML-Dateien usw.) zugewiesen.
- Mit Hilfe von Variablen können Datenelemente einer Sequenz mit anderen Datenelementen derselben Sequenz verglichen werden.
- Mit Hilfe von Variablen können Zwischensequenzen erstellt werden. Datensätze können gefiltert werden, bevor sie an eine Zielkomponente übergeben werden oder sie können nach der Variablen z.B. mit Hilfe der position-Funktion gefiltert werden.

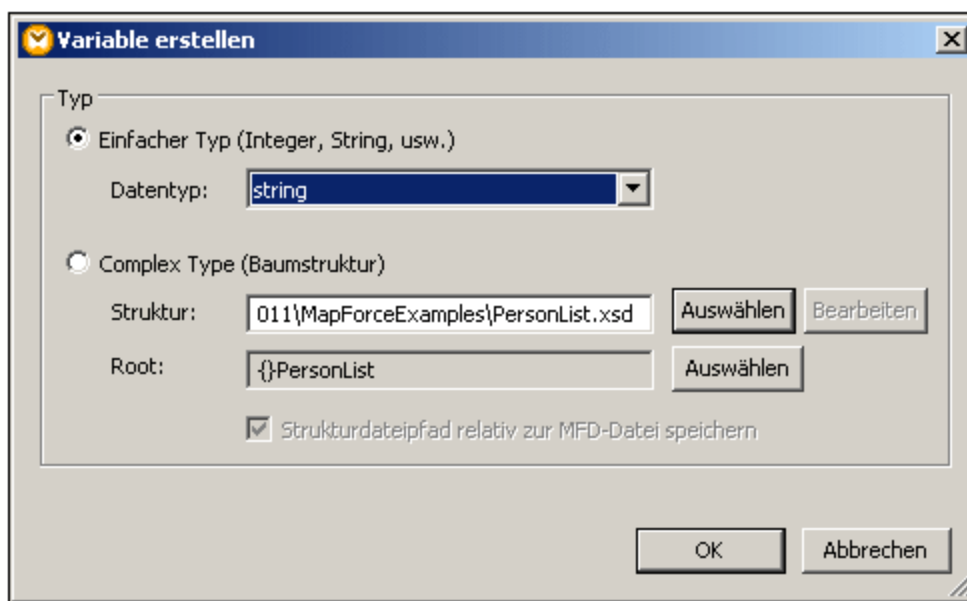
Unterschied zwischen Variablen und verketteten Mappings

Verkettete Mappings	Variablen
Erfolgen in zwei voneinander unabhängigen Schritten.	Werden je nach Kontext / Geltungsbereich ausgewertet. Werden durch eine "compute-when"-Verbindung gesteuert
Die Zwischenergebnisse werden bei der Ausführung des Mappings extern in XML-Dateien gespeichert.	Die Zwischenergebnisse werden bei der Ausführung des Mappings intern gespeichert und nicht in physischen Dateien.
Eine Vorschau des Zwischenergebnisses kann mit Hilfe der Vorschau-Schaltfläche angezeigt werden.	Das Ergebnis der Variablen kann nicht in einer Vorschau angezeigt werden.

So fügen Sie eine Zwischenvariablenkomponente ein:

Es gibt verschiedene Methoden um Zwischenvariablen einzufügen: Über die Menüoption, durch Klicken auf die Schaltfläche "Var." oder durch Rechtsklick auf die Input/Output-Symbole und Erstellen von Variablen auf Basis der Input-/Output-Komponente.

1. Wählen Sie die Menüoption **Einfügen | Variable...** oder klicken Sie in der Symbolleiste auf die Schaltfläche "Variable" . Sie können jetzt wählen, ob Sie eine Variable vom Typ "simple" oder "complex" einfügen möchten.

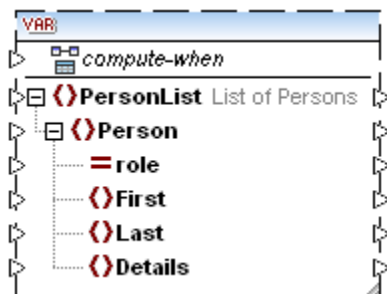


2. Aktivieren Sie das Optionsfeld für die Art der einzufügenden Variablen z.B. "Simple type" oder "Complex type".

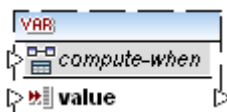
Wenn Sie das Optionsfeld "Complex type" aktiviert haben:

3. Klicken Sie auf die Schaltfläche "Auswählen" zum z.B. XML-Schema auszuwählen und wählen Sie im nächsten Dialogfeld das Root-Element aus.
4. Klicken Sie auf OK, um die Variable einzufügen.

Variable vom Typ "complex":



Variable vom Typ "simple":



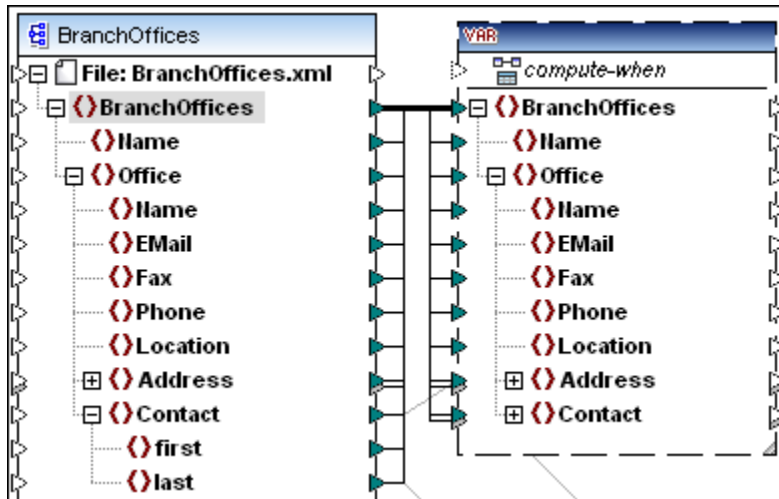
Eine solche Variable hat ein einziges mapbares Datenelement / einen einzigen Wert, z. B. string, integer. Beachten Sie, dass der "Wert" [dupliziert werden kann](#).

Alternative Methoden zum Einfügen von Variablen:

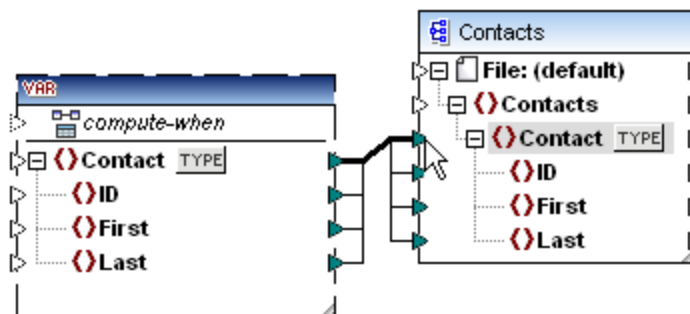
- Klicken Sie mit der rechten Maustaste auf ein **Output**-Symbol einer Komponente (z.B. BranchOffices) und wählen Sie "Variable anhand von Quell-Node erstellen".



Daraufhin wird anhand desselben Quellschemas eine Variable vom Typ "complex" erstellt und alle Datenelemente werden automatisch mit einer "Alles kopieren"-Verbindung verbunden.

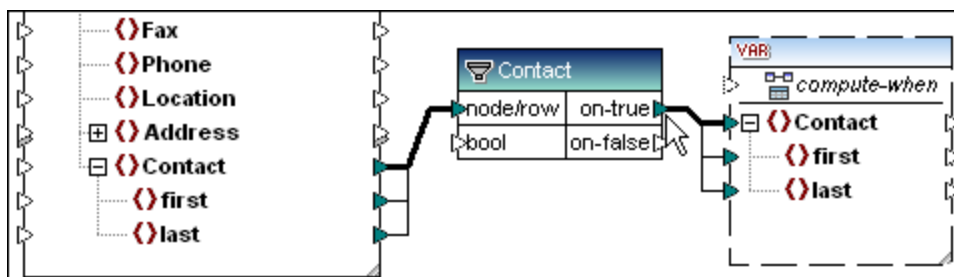


- Klicken Sie mit der rechten Maustaste auf ein **Input** -Symbol einer Zielkomponente (z. B. Contact) und wählen Sie den Befehl "Variable für Ziel-Node erstellen".



Daraufhin wird anhand desselben Quellschemas, das auch in von Zielkomponente verwendet wird, eine Variable vom Typ "complex" mit dem Datenelement "Contact" als Root-Node erstellt und alle Datenelemente werden automatisch mit einer "Alles kopieren"-Verbindung verbunden.

- Klicken Sie mit der rechten Maustaste auf das **Output**-Symbol einer Filter-Komponente (on-true/on-false) und wählen Sie den Befehl "Variable anhand von Quell-Node erstellen".



Daraufhin wird anhand desselben Quellschemas eine Komponente vom Typ "complex" erstellt und das mit dem Filter-Input-Node/ der Filter-Input-Zeile verbundene Datenelement, also Contact, wird automatisch als Root-Element der Zwischenkomponente verwendet.



Compute-when

Über das Input-Datenelement "compute-when" können Sie den **Geltungsbereich** der Variablen festlegen; anders ausgedrückt: Sie können festlegen, wann und wie oft der Wert der Variablen bei der Ausführung des Mappings berechnet wird. Sie müssen diesen Input nicht in vielen Fällen verbinden, es kann aber notwendig sein, um den Standard-Kontext außer Kraft zu setzen oder die Leistung des Mappings zu optimieren.

Im folgenden Text wird als **Substruktur** eine Gruppe von Datenelemente/Nodes in einer Zielkomponente mit allen ihren Nachfahren bezeichnet: z.B. ein Element <Person> mit seinen Child-Elementen <FirstName> und <LastName>.

Als **Variablenwert** werden die Daten bezeichnet, die auf der **Output**-Seite der Variablenkomponente zur Verfügung stehen.

- Bei "simple" Variablen handelt es sich hierbei um eine Sequenz atomarer Werte, die den in den Komponenteneigenschaften definierten Datentyp haben.
- Bei "complex" Variablen handelt es sich um eine Sequenz von Root Nodes (vom in den Komponenteneigenschaften definierten Typ), von denen jeder alle seine Nachfahren-Nodes enthält.

Die Sequenz atomarer Werte (oder Nodes) darf natürlich nur ein oder sogar null Datenelemente enthalten. Dies hängt davon ab, was mit der Input-Seite der Variablenkomponente und allen Parent-Datenelementen in der Quell- und Zielkomponente verbunden ist.

Compute-when - nicht verbunden (Standardeinstellung)

Wenn das compute-when-Input-Datenelement **nicht** (mit einem Output Node einer Quellkomponente) verbunden ist, wird der Variablenwert berechnet, sobald er **das erste Mal** in einer **Zielsubstruktur** (entweder über einen Konnektor von der Variablenkomponente direkt zu einem Node in der Zielkomponente oder indirekt über Funktionen) verwendet wird. Derselbe Variablenwert wird auch für alle Child-Ziel-Nodes innerhalb der Substruktur verwendet.

Der tatsächliche Wert der Variablen hängt von etwaigen Verbindungen zwischen übergeordneten Datenelementen der Quell- und Zielkomponente ab (siehe "[Schleifen, Gruppen und Hierarchien - der Kontext](#)").

Dieses Standardverhalten ist dasselbe wie das von "complex" Outputs von [regulären benutzerdefinierten Funktionen](#) und Webservice-Funktionsaufrufen.

Wenn die Variable mit mehreren **nicht miteinander in Zusammenhang stehenden** Ziel-Nodes verbunden ist, wird der Wert der Variablen für jeden davon separat berechnet. Dabei können in jedem Fall unterschiedliche Ergebnisse erzeugt werden, da unterschiedliche Parent-Verbindungen sich auf den Kontext auswirken, in dem der Wert der Variablen ausgewertet wird.

Compute-when - verbunden

Wenn das compute-when-Datenelement mit einem **Output Node** einer Quellkomponente verbunden ist, wird die Variable immer dann berechnet, wenn das **Quelldatenelement** in der Zielsubstruktur zum **ersten** Mal verwendet wird.

Die Variable verhält sich so, als wäre sie ein Child-Element des mit dem compute-when verbundenen Datenelements.

Auf diese Art kann die Variable an ein bestimmtes Quelldatenelement **gebunden** werden, d.h. die Variable wird zur Laufzeit immer dann neu ausgewertet, wenn ein neues Datenelement aus der Sequenz in der Quellkomponente gelesen wird.

Dies steht im Zusammenhang mit der allgemeinen Regel zu Verbindungen in MapForce - "erstelle für jedes Quelldatenelement ein Zieldatenelement". Bei compute-when bedeutet das "berechne für jedes Quelldatenelement den Variablenwert".

Compute-once

Wenn Sie mit der rechten Maustaste auf das "compute-when"-Symbol klicken und im Kontextmenü den Befehl "Einmal berechnen" auswählen, ändert sich das Symbol in "compute-when=once" und das Input-Symbol wird entfernt.

Bei dieser Einstellung wird der Variablenwert nur einmal berechnet, und zwar **vor** allen Werten der Zielkomponenten, sodass die Variable praktisch zu einer globalen Konstante für das restliche Mapping wird.

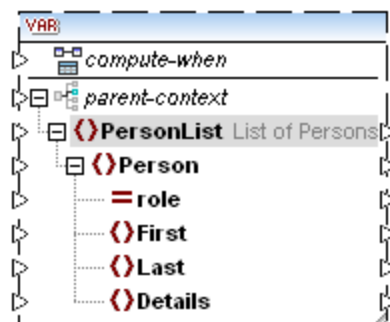
In einer benutzerdefinierten Funktion wird die Variable bei jedem Funktionsaufruf ausgewertet, bevor das tatsächliche Funktionsergebnis ausgewertet wird.

Parent-Kontext

Hauptsächlich wird ein Parent-Kontext hinzugefügt, wenn mehrere Filter verwendet werden, und Sie einen zusätzlichen Parent-Node benötigen, über den iteriert werden kann.

So fügen Sie einen Parent-Kontext zu einer Variablen hinzu:

- Klicken Sie mit der rechten Maustaste auf den Root Node, z.B. PersonList und wählen Sie im Kontextmenü den Befehl "Parent-Kontext hinzufügen". Daraufhin wird ein neuer Node hinzugefügt "parent-context" zur bestehenden Hierarchie hinzugefügt.

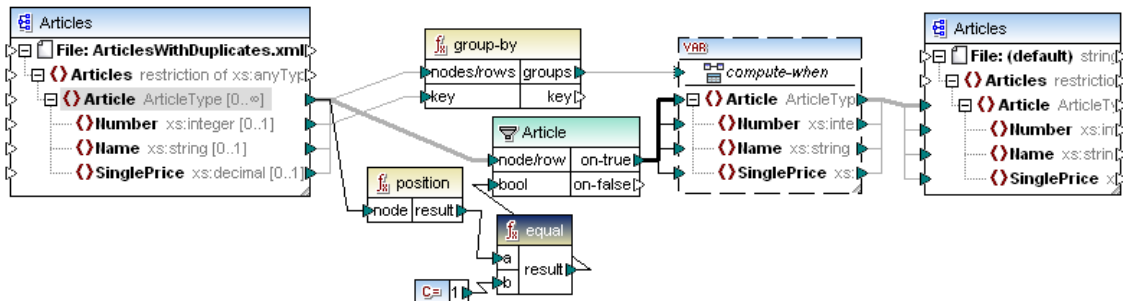


Durch den Parent-Kontext wird innerhalb der Komponente ein virtueller übergeordneter (Parent) Node zur Hierarchie hinzugefügt. Auf diese Art können Sie über einen zusätzlichen Node in der Quellkomponente iterieren.

11.1 Variablen - Anwendungsfälle

Herausfiltern mehrerer Instanzen desselben Datensatzes aus einer Quellinstanz

Quelldaten können oft mehrere Instanzen desselben Datensatzes enthalten. In den meisten Fällen soll nur einer dieser Datensätze auf die Zielkomponente gemappt werden. Das Beispiel unten finden Sie unter dem Namen **DistinctArticles.mfd** im Ordner ...\\MapForceExamples.



Die Datei `ArticlesWithDuplicates.xml` enthält zwei Artikel mit derselben Artikelnummer (zwei mit der Artikelnummer 1 und zwei mit der Artikelnummer 3).

Articles

xmlns:xsi

http://www.w3.org/2001/XMLSchema-instance

xsi:noNamespace

Articles.xsd

Article (6)

	Number	Name	SinglePrice
1	1	T-Shirt	25
2	1	T-Shirt Duplicate	25
3	2	Socks	2.30
4	3	Pants	34
5	4	Jacket	57.50
6	3	Pants Duplicate	34

Die Artikelnummer wird als Schlüssel in der [group-by](#)-Funktion verwendet, sodass damit eine Gruppe pro (eindeutiger) Artikelnummer erstellt wird. Daher enthält jede Gruppe einen Artikel und alle unerwünschten Duplikate dieses Artikels. Der nächste Schritt ist, das erste Datenelement jeder Gruppe zu extrahieren und den Rest zu löschen.

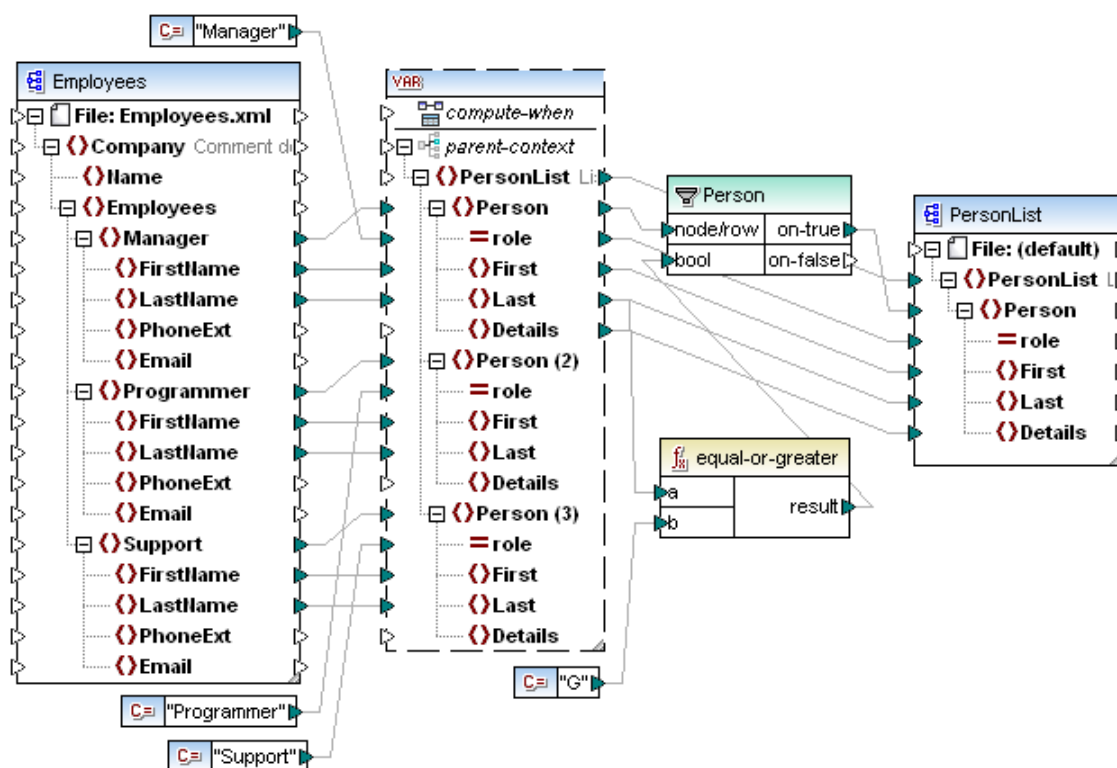
Durch die Verbindung der Gruppierungsausgabe mit `compute-when` wird die Variable einmal für jede Gruppe im Kontext dieser Gruppe ausgewertet. Auf diese Weise wird eine zusätzliche Kontextebene gebildet, so als hätten wir ein übergeordnetes Element des Article-Zielelements verbunden.

Um den ersten Artikel jeder Gruppe auszuwählen, verwenden wir die `position`-Funktion und eine Filter-Komponente, die mit der Input-Seite der Variablen verbunden wird.

Anwenden von Filtern auf Zwischensequenzen:

Nodes in Variablenkomponenten können wie bei jeder anderen Komponentenart [dupliziert](#) werden. Auf diese Art können Sie Sequenzen aus mehreren unterschiedlichen Quellen bilden und diese Sequenz dann weiterverarbeiten.

So sehen Sie z.B. in der unten stehenden Abbildung, wie `PersonList.mfd` mit Hilfe einer Zwischenvariablen geändert werden könnte und wie auch Konstantenkomponenten als Quelldatenelemente fungieren können.,

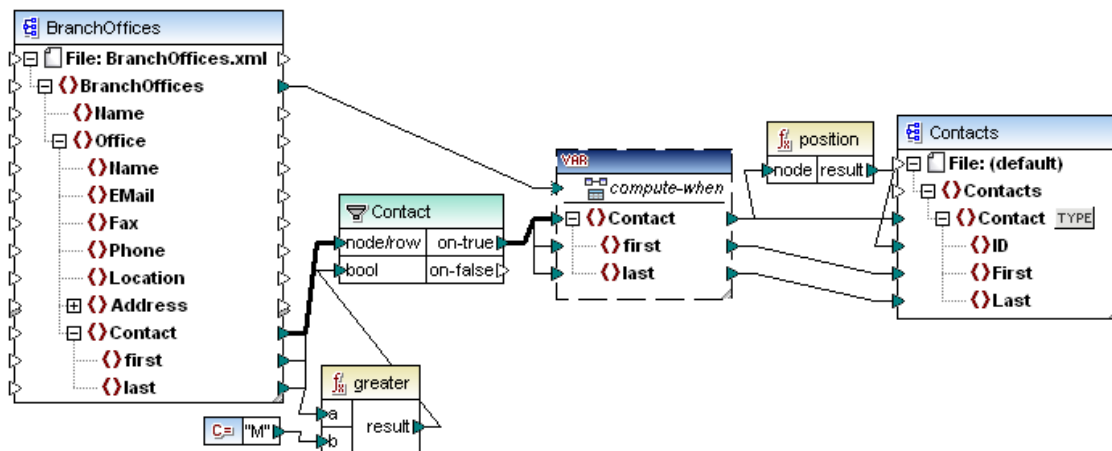


Der Node "Person" der Variable wurde zwei Mal kopiert und es wurde ein Filter hinzugefügt, um Personen, deren Nachname (LastName) mit einem Buchstaben nach "G" beginnt, zu filtern.

Nummerieren von Nodes einer gefilterten Sequenz

Das Beispiel unten finden Sie unter dem Namen **PositionInFilteredSequence.mfd** im Ordner ...MapForceExamples. In diesem Beispiel werden mit Hilfe der Variablen die gefilterten Kontakte gesammelt, deren Nachname mit einem Buchstaben nach M beginnt.

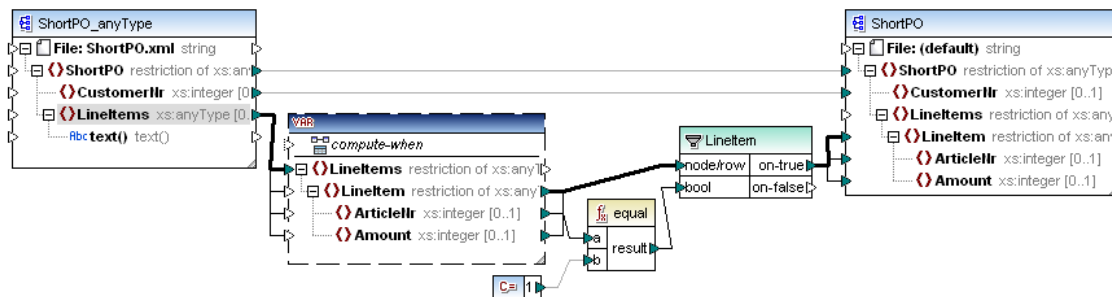
Die Kontakte werden anschließend an die Zielkomponente übergeben, wobei diese Kontakte mit Hilfe der position-Funktion der Reihe nach nummeriert werden.



Extrahieren bestimmter Daten aus dem anyType-Node einer Quellkomponente

Dieses Beispiel besteht aus einer Quellkomponente, die anyType-Elemente enthält, aus denen

wir bestimmte Daten filtern möchten.



Die Zwischenvariable basiert auf einem Schema, das Nodes desjenigen Datentyps hat, den wir mappen möchten, d.h. ArticleNr und Amount sind beide vom Typ Integer. Diese spezifischen Daten werden von ArticleNr. gefiltert und an die Zielkomponente übergeben.

Kapitel 12

Digitale Signaturen

12 Digitale Signaturen

Digitale Signaturen sind in einer W3C-Spezifikation definiert und dienen dazu ein XML-Dokument mit einem verschlüsselten Code digital zu signieren. Anhand dieses Codes kann überprüft werden, ob das XML-Dokument nach der Signierung geändert wurde. Die XML-Signaturfunktion in MapForce unterstützt nur Zertifikate vom Typ RSA-SHA1 und DSA-SHA1.

Nähere Informationen zu XML-Signaturen finden Sie in der W3C-Spezifikation für XML-Signaturen unter <http://www.w3.org/TR/xmlsig-core/>

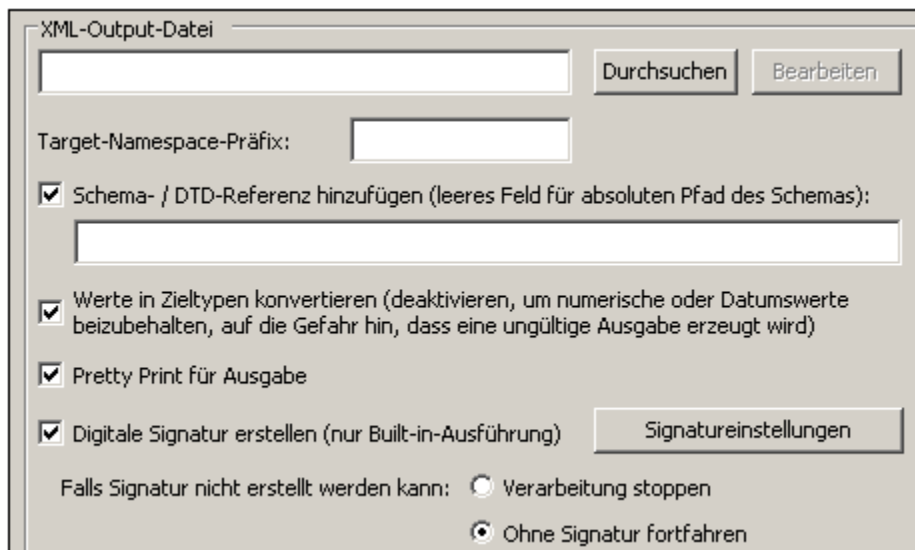
MapForce unterstützt die Erstellung digitaler XML-Signaturen für XML- und XBRL-Ausgabedateien. Digitale Signaturen können nur generiert werden, wenn das Ausgabeziel BUILTIN ist. Wenn die Schaltfläche "Ausgabe" betätigt und die Ergebnisdatei gespeichert wird, so wird eine Signatur für die generierte Ergebnisdatei erstellt.

Digitale Signaturen können als letztes Element in das Ausgabedokument eingebettet oder in einer separaten Datei gespeichert werden.

- Bei Auswahl der Option "Enveloped" ist die Signatur das letzte Child-Element unterhalb des Root-Elements der XML-Datei.
- Bei Auswahl der Option "Detached" wird die Signaturdatei als separates Dokument generiert.

So aktivieren Sie die Generierung digitaler Signaturen

1. Öffnen Sie das Dialogfeld "Komponenteneinstellungen" der Ausgabekomponente, durch Doppelklick auf die Titelleiste der Komponente oder durch Auswahl des Befehls **Komponente | Eigenschaften**.
2. Aktivieren Sie das Kontrollkästchen "Digitale Signaturen erstellen".



3. Daraufhin wird das Dialogfeld [XML-Signatureinstellungen](#) automatisch geöffnet.
4. Geben Sie die Einstellungen ein und klicken Sie auf OK.

So ändern Sie die Einstellungen für digitale Signaturen:

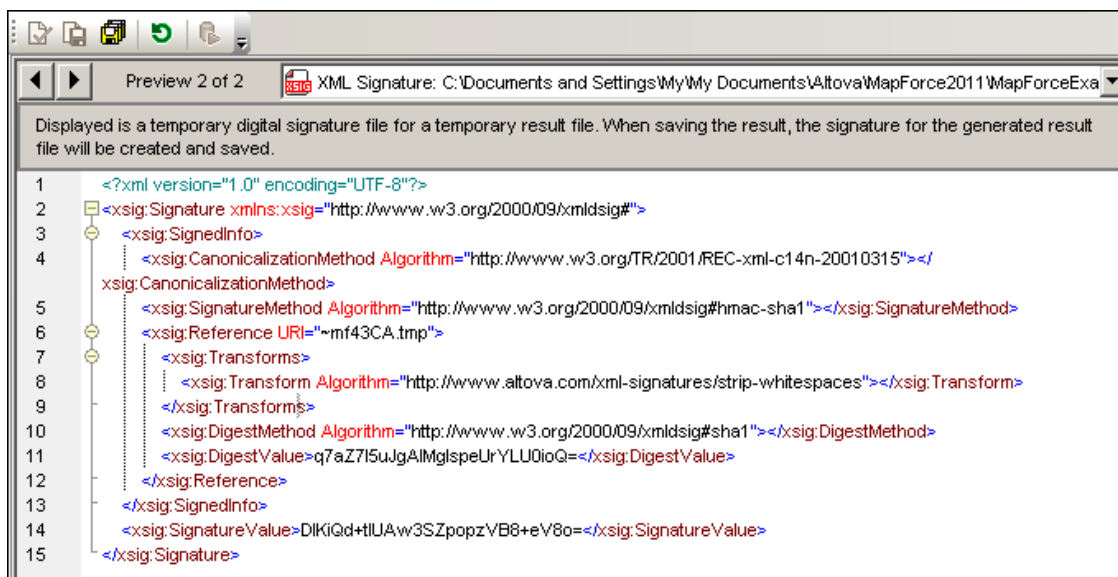
1. Öffnen Sie das Dialogfeld "Komponenteneinstellungen" der Ausgabekomponente, durch Doppelklick auf die Titelleiste der Komponente.
2. Klicken Sie auf die Schaltfläche "Signatureinstellungen" um das Dialogfeld [XML-Signatureinstellungen](#) zu öffnen.
3. Geben Sie die Einstellungen ein und klicken Sie auf OK.

Wir wollen als Beispiel die Datei **MarketingAndDailyExpenses.mfd** aus dem Ordner ... \MapForceExamples verwenden:

1. Doppelklicken Sie auf die Zielkomponente "MarketingExpenses" und anschließend auf die Schaltfläche "Signatureinstellungen".
2. Wählen Sie die Optionen wie in der Abbildung oben gezeigt aus und klicken Sie auf OK um das Dialogfeld zu schließen.,
3. Klicken Sie auf die Ausgabe-Schaltfläche, um das Mapping-Ergebnis zu sehen.

Im Vorschauenfenster werden zwei Dateien generiert: MarketingExpenses.xml ist das Mapping-Ergebnis der Zielkomponente.

MarketingExpenses.xml.xsig (unten gezeigt) ist die temporäre Signaturdatei, die zur Zielkomponente generiert wird.



So generieren Sie die Signaturdatei:

1. Klicken Sie in der Symbolleiste der Ausgabevorschau auf die Schaltfläche "Alle generierten Ausgaben speichern"  .
Daraufhin werden im Ausgabeverzeichnis die .XML- und die .XSIG-Datei generiert.

In der Datei **MarketingExpenses_EnvelopedSignature.mfd** aus dem Ordner ...
 \MapForceExamples sehen Sie das Ergebnis, wenn die Signaturplatzierung "enveloped" erfolgt.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <marketing-expenses xmlns:xsig="http://www.w3.org/2000/09/xmldsig#" xsi:noNamespaceSchemaLocation="
   C:/DOCUME~1/MYDOCU~1/Altova/MapForce2011/MapForceExamples/MarketingExpensesSigned.xsd" xmlns:xsi="
   http://www.w3.org/2001/XMLSchema-instance">
3    <Person>
4      <FullName>Fred Landis</FullName>
5      <Title>Project Manager</Title>
6      <Phone>123-456-7890</Phone>
7      <Email>f.landis@nanonull.com</Email>
8    </Person>
9    <expense-item>
10     <type>Meal</type>
11     <Date>2003-01-01</Date>
12     <expense>122.11</expense>
13   </expense-item>
14   <expense-item>
15     <type>Lodging</type>
16     <Date>2003-01-02</Date>
17     <expense>299.45</expense>
18   </expense-item>
19   <xsig:Signature>
20     <xsig:SignedInfo>
21       <xsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
22       <xsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"/>
23       <xsig:Reference URI="">
24         <xsig:Transforms>
25           <xsig:Transform Algorithm="http://www.altova.com/xml-signatures/strip-whitespaces"/>
26           <xsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
27         </xsig:Transforms>
28         <xsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
29         <xsig:DigestValue>HvI5SnMsLf8RBsgy7fIj5+leD0M=</xsig:DigestValue>
30       </xsig:Reference>
31     </xsig:SignedInfo>
32     <xsig:SignatureValue>a+uml6qhJ1SN1J00/HKV7h99Y6g=</xsig:SignatureValue>
33   </xsig:Signature>
34 </marketing-expenses>

```

Gültigkeit des XML-Dokuments

Wenn eine XML-Signatur in das XML-Dokument eingebettet wird, so wird ein Signature Element aus dem Namespace `http://www.w3.org/2000/09/xmldsig#` zum XML-Dokument hinzugefügt. Damit das Dokument dem Schema gemäß gültig bleibt, muss das Schema die entsprechenden Elementdeklarationen enthalten. MapForce bettet Signaturen mit der Option "Enveloped" ein:

- *Enveloped*: Das Element `Signature` wird als das letzte Child-Element des Root (Dokument)-Elements erstellt.

Wenn Sie das Schema des XML-Dokuments nicht ändern möchten, kann die XML-Signatur mit der Option "detached" in einer externen Datei erstellt werden.

Unterhalb sehen Sie Auszüge aus XML-Schemas, in denen gezeigt wird, wie das Signature Element einer "enveloped" Signatur verwendet werden darf. Sie können diese Beispiele als Richtlinien für die Anpassung Ihrer eigenen Schemas verwenden.

In ersten der beiden Beispiele unten wird das XML-Signaturschema in das Benutzerschema importiert. Das XML-Signatur-Schema befindet sich unter der Internetadresse:

<http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd>

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:xsig="http://www.w3.org/2000/09/xmldsig#"
            elementFormDefault="qualified"
            attributeFormDefault="unqualified">
  <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
            schemaLocation="
http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
  <xs:element name="Root">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="FirstChildOfRoot"/>
        <xs:element ref="SecondChildOfRoot" minOccurs="0"/>
        <xs:element ref="ThirdChildOfRoot" minOccurs="0"/>
        <xs:element ref="xsig:Signature" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  ...
</xs:schema>

```

Die zweite Option (*Beispiel unten*) ist, ein allgemeines Wildcard-Element hinzuzufügen, das jedes Element aus anderen Namespaces repräsentieren kann. Wenn Sie das Attribut `processContents` auf `lax` setzen, überspringt der Validator dieses Element, da keine übereinstimmende Elementdeklaration gefunden wird. Folglich muss der Benutzer das XML-Signatur-Schema nicht referenzieren. Der Nachteil dieser Option ist allerdings, dass an der angegebenen Stelle im XML-Dokument jedes beliebige Element (und nicht nur das Element `Signature`) hinzugefügt werden kann, ohne dass das XML-Dokument ungültig wird.

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified"
            attributeFormDefault="unqualified">
  <xs:element name="Root">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="selection"/>
        <xs:element ref="newsitems" minOccurs="0"/>
        <xs:element ref="team" minOccurs="0"/>
        <xs:any namespace="##other" minOccurs="0" processContents="lax"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  ...
</xs:schema>

```

12.1 XML-Signatureeinstellungen

XML-Signatureeinstellungen

XML-Signatureeinstellungen

☐ Zertifikat:

☒ Passwort:

Das Passwort muss mindestens 5 und maximal 16 Zeichen lang sein.

☐ Passwort in MFD-Datei speichern

Transformationen

☒ Whitespaces zwischen XML-Elementen entfernen

Dies ist eine Altova-spezifische Transformationsfunktion, wodurch XML-Signaturen zulässige Whitespace-Änderungen besser berücksichtigen, sodass die Signatur eines Dokuments bei einem Wechsel zwischen der Text-, der Grid- und der Authentic-Ansicht im Altova XMLSpy Editor nicht ungültig wird. Es werden nur Whitespace-Zeichen entfernt, die sich außerhalb des Attribut- oder Element-Inhalts befinden und somit keine Bedeutung haben.

☒ Keine

☐ Canonical XML 1.0

☐ Canonical XML 1.0 mit Kommentaren

☐ Canonical XML 1.1

☐ Canonical XML 1.1 mit Kommentaren

☐ Base64

Platzierung der Signatur

☐ Enveloped: Die Signatur wird als das letzte Child des Root-Elements in den bestehenden XML-Code eingefügt

Hinweis: Die Platzierung der XML-Signatur an dieser Stelle muss gemäß Ihrem XML-Schema gestattet sein, da das XML-Dokument sonst ungültig wird. Informationen, wie Sie Ihr Schema entsprechend modifizieren können, finden Sie in der Dokumentation.

☒ Detached: Die Signatur wird in einer eigenen Datei gespeichert

Signaturdatei erstellen

Mit Dateierweiterung

☒ Signatur-Dateierweiterung an Dateinamen anhängen

☐ Vorhandene Dateierweiterung durch Signaturerweiterung ersetzen

☒ Relativen Pfad zur signierten Datei in Detached Signature-Datei verwenden

☒ KeyInfo anhängen

Die Signatureinstellungen werden für jede Komponente einzeln im Dialogfeld "Komponenteneinstellungen" der jeweiligen Komponente gespeichert. Alle diese Einstellungen werden beim Speichern in der MFD-Datei gespeichert.

Authentifizierungsmethode: Zertifikat oder Passwort

Die Signatur kann auf einem Zertifikat oder einem Passwort basieren. Wählen Sie das Optionsfeld für die gewünschte Methode aus.

- **Zertifikat:**
Wenn Sie ein Zertifikat verwenden möchten, muss dieses einen privaten Schlüssel (private key) haben und sich in einem Zertifikatspeicher befinden, auf den Zugriff besteht. Die Signatur wird anhand des privaten Schlüssels des Zertifikats generiert. Zur Überprüfung der Signatur muss Zugriff auf das Zertifikat (oder eine Public-Key-Version davon) bestehen. Der öffentliche Schlüssel (public key) des Zertifikats dient zur

Überprüfung der Signatur. Klicken Sie zur Auswahl des Private-Public-Key-Zertifikats auf die Schaltfläche **Auswählen** und navigieren Sie zum Zertifikat.

- *Passwort:* Geben Sie ein Passwort mit einer Länge von fünf bis 16 Zeichen ein. Anhand dieses Passworts wird die Signatur erstellt und dieses Passwort wird später benötigt, um die Signatur zu überprüfen. Die Schaltfläche "OK" des Dialogfelds wird nur aktiv, wenn diese Anforderung erfüllt wird.
- *Passwort in MFD-Datei speichern:*
Wenn diese Option aktiv ist, wird das in das Feld "Passwort" eingegebene Passwort in verschlüsselter Form in der MFD-Datei gespeichert, so dass es für Personen nicht lesbar ist. Beachten Sie, dass jeder Benutzer, der Zugriff auf die MFD-Datei hat, mit Hilfe dieses Passworts Signaturen erstellen kann.

Transformationen

Die XML-Daten werden transformiert und das Ergebnis der Transformation wird zur Erstellung der Signatur verwendet. Sie können den Kanonisierungsalgorithmus, der auf die XML-Daten der Datei (den SignedInfo-Inhalt) anzuwenden ist, definieren, bevor die Signatur berechnet wird. Im Folgenden einige wichtige Unterschiede zwischen den Algorithmen:

- *Canonical XML mit oder ohne Kommentare:*
Wenn Kommentare bei der Berechnung der Signatur einbezogen werden, so führt jede Änderung an den Kommentaren in den XML-Daten zu einem Fehler bei der Überprüfung der Signatur. Im anderen Fall dürfen Kommentare nach der Signierung des Dokuments geändert oder hinzugefügt werden, ohne dass die Signatur dadurch ungültig wird.

Anmerkung:

"...mit Kommentaren" steht nur für die Platzierung mit der Option "detached" zur Verfügung.

- *Base64:*
Das Root- (oder Dokument-) Element des XML-Dokuments wird als Base64-kodiert betrachtet und in seiner Binärform gelesen. Wenn das Root-Element nicht in Base64 kodiert ist, wird ein Fehler zurückgegeben oder das Element als leer gelesen, je nachdem, welcher Elementtyp verwendet wird.
- *Keine:*
Es wird keine Transformation durchgeführt und die XML-Daten aus der auf der Festplatte gespeicherten Binärdatei werden direkt an die Signaturüberprüfung übergeben. Alle späteren Änderungen an den Daten führen dazu, dass die Signatur nicht verifiziert werden kann.

Wenn allerdings das Kontrollkästchen **Whitespaces zwischen XML-Elementen entfernen** aktiviert ist, so werden alle Whitespace-Zeichen entfernt und Änderungen an Whitespace-Zeichen werden bei der Signaturüberprüfung ignoriert.

Ein grundlegender Unterschied zwischen der Option *Keine* und einer der *Canonical*-Optionen ist, dass bei der Kanonisierung ein XML-Datenstrom erzeugt wird, in dem einige Unterschiede wie z.B. die Reihenfolge der Attribute normalisiert wird. Als Folge davon werden Änderungen wie eine andere Attributreihenfolge bei einer Kanonisierungstransformation normalisiert (sodass die Signatur verifiziert werden kann), während eine solche Änderung berücksichtigt wird, wenn keine Transformation durchgeführt wird (d.h. die Signatur kann nicht verifiziert werden).

Beachten Sie allerdings, dass standardmäßig eine Kanonisierung durchgeführt wird, wenn die Signatur (mit der Option "enveloped" eingebettet. D.h. die XML-Daten werden unverändert verwendet (d.h. ohne transformiert zu werden), wenn die Signatur "detached" also separat ist, wenn *Keine* ausgewählt ist und wenn das Kontrollkästchen *Whitespaces entfernen* deaktiviert ist.

Platzierung der Signatur

Die Signatur kann in die XML-Datei platziert oder als separate Datei erstellt werden. Es stehen die folgenden Optionen zur Verfügung:

- *Enveloped:*
Das Signaturelement wird als das letzte Child-Element des Root (Dokument)-Elements erstellt. Achtung: Das mit der Datei verknüpfte XML-Schema muss die Elemente zur Definition der Signatur enthalten, damit die XML-Ausgabedatei gültig ist. Nähere Informationen dazu finden Sie ganz oben in diesem Abschnitt.
- *Detached:*
Die XML-Signatur wird in Form einer separaten Datei erstellt. In diesem Fall können Sie die Dateierweiterung der Signatur definieren und festlegen ob: (i) die Erweiterung an den Namen der XML-Datei angehängt werden soll (z.B. `test.xml.xsig`) oder (ii) ob die Erweiterung die XML-Erweiterung der Datei ersetzen soll (z.B. `test.xsig`). Außerdem können Sie festlegen, ob die Referenz auf die XML-Datei in der Signaturdatei ein relativer oder absoluter Pfad sein soll.

Anmerkung: XML-Signaturen für XML-Schema- (.xsd) und XBRL-Dateien können nur als externe Signaturdateien erstellt werden. Signaturen für WSDL-Dateien können als externe Dateien erstellt und über die Option "Enveloped" in die WSDL-Datei inkludiert werden.

Anmerkung: Wenn die XML-Signatur als separate Datei erstellt wird, werden die XML-Datei und die Signaturdatei durch eine Referenz in der Signaturdatei miteinander verknüpft. Folglich muss, wenn sich die Signatur in einer externen Datei befindet, bei der Überprüfung der Signatur die Signaturdatei aktiv sein und nicht die XML-Datei.

Schlüsselinformationen anhängen

Die Option *KeyInfo anhängen* steht zur Verfügung, wenn die Signatur auf einem Zertifikat basiert. Wenn die Signatur auf einem Passwort basiert, steht sie nicht zur Verfügung.

Wenn die Option aktiv ist, werden die Daten des öffentlichen Schlüssels in die Signatur platziert. Im anderen Fall werden die Schlüsselinformationen nicht in die Signatur inkludiert. Der Vorteil bei der Inkludierung der Schlüsselinformationen ist, dass das Zertifikat selbst (insbesondere die Daten des öffentlichen Schlüssels) für die Überprüfung nicht benötigt werden (da die Schlüsselinformationen ja in der Signatur vorhanden sind).

Signatureinstellungen - ungültig

MapForce kann eine Ausgabe nicht mit einer digitalen Signatur versehen, wenn die Signatureinstellungen ungültig sind. Die Signatureinstellungen sind ungültig wenn:

- kein Zugriff auf das ausgewählte Zertifikat besteht oder es sich nicht zum Signieren von XML-Dokumenten eignet oder
- kein Passwort definiert wurde, z.B. die Option "Passwort in mfd-Datei speichern" nicht aktiviert wurde.

Wenn Sie auf die Ausgabe-Schaltfläche klicken, werden Sie von MapForce folgendermaßen nach dem Passwort gefragt:

Geben Sie bitte ein Passwort für die Signierung der Ausgabe der Komponente "MarketingExpensesSigned" an

und folgendermaßen nach dem Zertifikat:

Wählen Sie bitte den Speicher aus, der das Zertifikat zum Signieren der Ausgabe der Komponente "MarketingExpensesSigned" enthält.

Wenn kein Passwort oder Zertifikat ausgewählt wird, wird die Verarbeitung entweder gestoppt oder ohne Signatur fortgesetzt. Sie können dies im Dialogfeld "Komponenteneinstellungen" über die Optionsfelder "Verarbeitung stoppen" oder "Ohne Signatur fortfahren" festlegen.

Wenn das Mapping über die Befehlszeile ausgeführt wird, erscheint kein solches Dialogfeld. Die Mapping-Ausführung wird entweder mit einem Fehler abgebrochen oder ohne Signatur fortgesetzt..

Kapitel 13

MapForce Tipps...

13 MapForce Tipps...

In diesem Abschnitt werden häufige Aufgabenstellungen beschrieben, wie sie beim Erstellen Ihrer eigenen Mapping-Projekte oft auftreten werden.

Beschrieben werden die folgenden Aufgaben:

Allgemeines:

Aufgabe:

Informationen in diesem Abschnitt:

Filtern von Daten anhand bestimmter Kriterien

[Filtern - Abrufen dynamischer Daten](#)

Sortieren von Input-Daten auf Basis eines bestimmten Schlüssels

[Sortierkomponente - Sortieren von Input-Sequenzen](#)

Verwenden von **dynamischen/mehreren Input- und Output-Dateien** beim Mapping

[Dynamische Input/Output-Dateien pro Komponente](#)

Mappen/Verwenden eines **derived complex type** (xsi:type)

[Derived XML Schematypen - Mappen](#)

Erstellen eines **rekursiven** benutzerdefinierten Mappings

[Rekursives benutzerdefiniertes Mapping](#)

Vorschau auf Mappings (vor der Generierung von Code)

[Mapping-Vorschau - Built-In-Ausführungsprozessor](#)

Verwendung von min, max, sum, avg und count-Aggregatsfunktionen

[Aggregatfunktionen: min, max, sum](#)

Verwenden einer **Input-Komponente** als **Parameter** bei einer Ausführung über die Befehlszeile

[Befehlszeile - Definieren von Input/Output-Dateien](#)

Definieren eines **Alternativwerts** für eine Input-Komponente

[Befehlszeile - Definieren von Input/Output-Dateien](#)

Definieren und Ausführen eines Mappings mit anderen Input- und Output-Dateien als den bei Erstellung des Mappings definierten

[Befehlszeile - Komponentennamen](#)

Transformieren eines Input-Werts mittels **Lookup-Tabelle** in einen Output-Wert

[Wertezuordnung - Transformieren von Input-Daten](#)

Informationen über die Typkonvertierung beim Mappen

[Überprüfung der Typkonvertierung](#)

Erstellen Ihrer eigenen **Katalogdateien**

[Katalogdateien in MapForce](#)

Zusammenführen mehrerer Quelldateien in einer einzigen Zieldatei

[Zusammenführe mehrerer Dateien in einer Zieldatei](#)

Nodes

Aufgabe

Informationen in diesem Abschnitt:

Testen von Nodes; vorhandene / nicht vorhandene Nodes

[Node-Tests, exists / not-exists](#)

Gruppieren von Nodes nach Ihrem Inhalt

[Gruppieren von Nodes / Node-Inhalt](#)

Mappen von Daten auf Basis der Position eines Node in einer Sequenz

[Position von Kontextelementen in einer Sequenz](#)

Kommentieren eines Mappings / bestimmter Konnektoren oder Nodes

[Annotationen / Kommentare](#)

Aufgabe

Informationen in diesem Abschnitt:

Erstellen eines gültigen XML-Zieldokuments mit einem Root-Element

[Mappings und das Root-Element von Zieldokumenten](#)

Sie möchten wissen, was bei Anwendung eines Filters an einem übergeordneten Element mit **Subelementen** passiert

[Filterkomponenten - Tipps](#)

Ersetzen **bestimmter (hex)-Zeichen** in einer Datenbank durch andere in der Zieldatei.

[Ersetzen bestimmter Zeichen in Datenbankdaten](#)

Definieren des Node, der als **Kontextnode** in einer Quelldatei verwendet werden soll.

Prioritätskontext
Node/Datenelement

Mappen mehrerer **hierarchischer Tabellen** auf eine einzige XML-Zieldatei

[Mappen mehrerer Tabellen auf eine XML-Datei](#)

Ausführen von MapForce über die **Befehlszeile (Ent/Pro)**

[Befehlszeilenparameter](#)

Definieren von **Ausnahmen** in Mappings

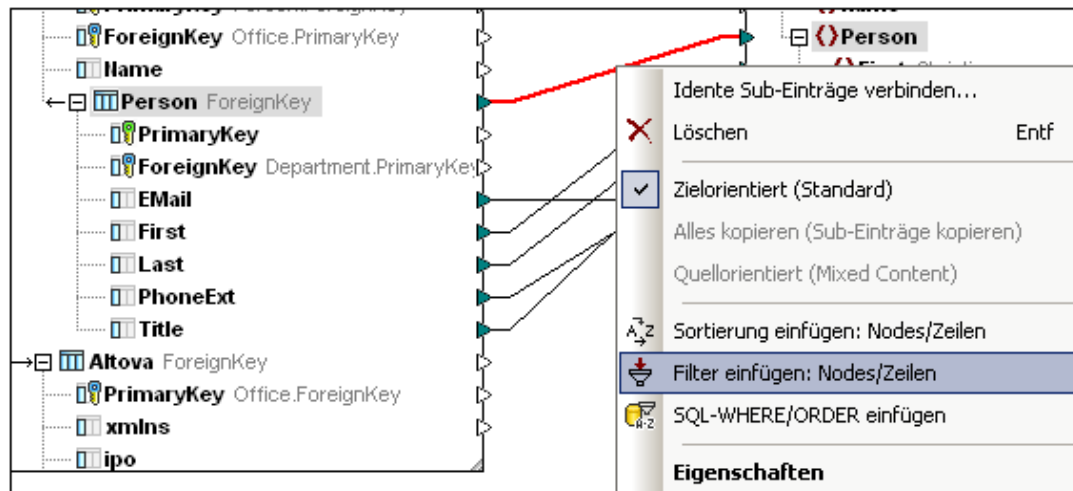
[Ausnahmeereignisse](#)

13.1 Filtern - Abrufen dynamischer Daten, Lookup-Tabelle

Wenn Sie Daten anhand bestimmter Kriterien abrufen/filtern möchten, verwenden Sie die **Filter**-Komponente.

So fügen Sie eine Filterkomponente ein:

1. Klicken Sie mit der rechten Maustaste auf einen Konnektor zwischen den zu filternden Nodes/Datenelementen.



2. Wählen Sie im Kontextmenü den Befehl **"Filter einfügen: Nodes/Zeilen"**. Daraufhin wird die Filterkomponente eingefügt und automatisch mit der Quell- und der Zielkomponente verbunden.
3. Definieren Sie die Bedingung, nach der gefiltert werden soll, z.B. `PrimaryKey=4` und verbinden Sie das Ergebnis mit dem bool-Parameter des Filters.

Alternativ dazu:

- Klicken Sie in der Symbolleiste auf das Filter-Symbol , um die Komponente einzufügen. Daraufhin wird die Filterkomponente ohne Verbindungslinien eingefügt. Dabei wird in der Titelleiste der Komponente der Titel "filter" angezeigt.

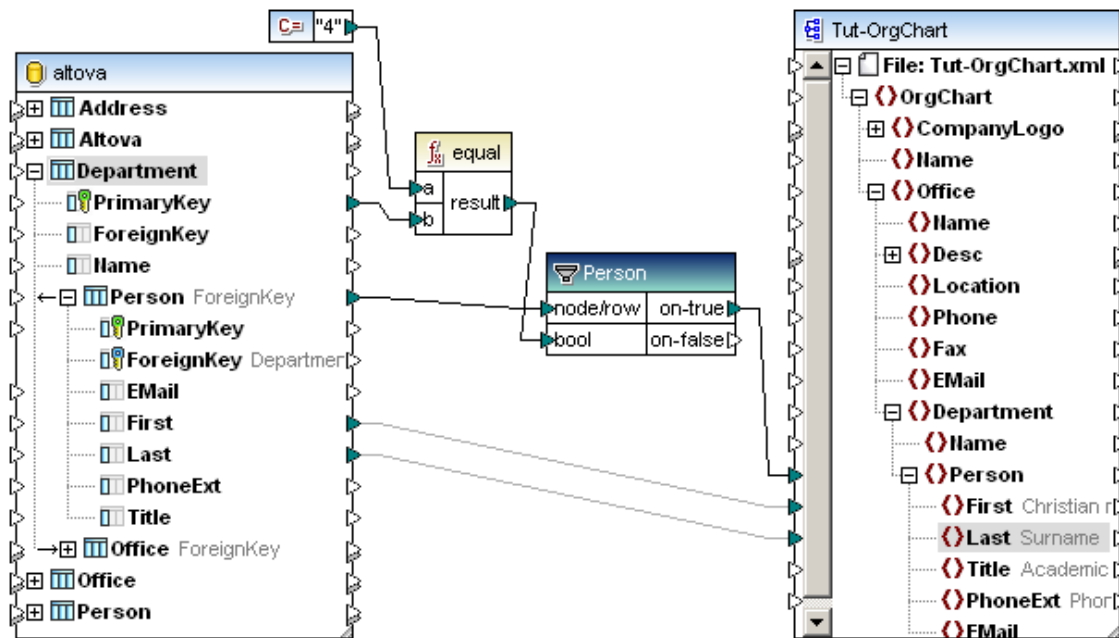


Sobald eine Verbindung zur Quellkomponente hergestellt wurde, wird in der Titelleiste der Name des Node/Zeileneintrags, der mit der Filter-Komponente verbunden wurde, in diesem Fall "Person" angezeigt.

Ziel: Mappen aller Vor- und Nachnamen (First, Last) aus der Tabelle **Person**, wo der **Primärschlüssel** (PrimaryKey) 4 ist.

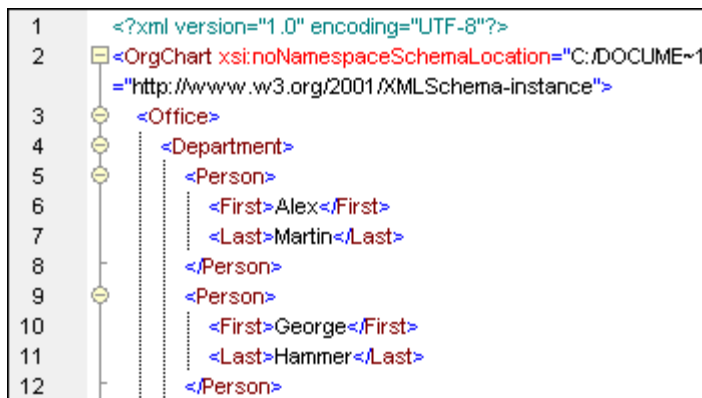
- Der Wert des Primärschlüssels wird mit dem Wert 4, der mit Hilfe der equal-Funktion von der Konstantenkomponente bereitgestellt wird, verglichen.
- Es wird der Eintrag **PrimaryKey** aus der "Root-Tabelle" "Department" gemappt.
- Aus der Tabelle "Person", die ein Child der Root-Tabelle "Department" ist, werden die Einträge **First** und **Last** gemappt.
- Die unter **derselben** "Root-Tabelle" (in diesem Fall "Department") definierten Mappings

behalten die Beziehungen zwischen ihren Tabellen bei.



Ein Filter hat zwei Input- und Output-Parameter: **node/row** und **bool** und **on-true**, **on-false**. Wenn die Boolesche Bedingung **true** ergibt, wird der Inhalt der mit dem **node/row**-Parameter verbundenen **Child**-Nodes der Quellkomponente an die Zielkomponente übergeben. Wenn der Boolesche Parameter **false** ergibt, können die Komplementwerte verwendet werden, indem eine Verbindung zum **on-false**-Parameter hergestellt wird.

Auf dem Register "Ausgabe" sehen Sie die Vor- und Nachnamen der Personen in der IT-Abteilung mit dem Primärschlüssel 4.



Weitere Beispiele zum Filtern von Daten finden Sie unter:

- XML-Daten filtern: [Filtern von Daten](#):
- Filtern von CSV-Dateien nach Kopfzeile und Details: [Hierarchien für CSV-Dateien und Dateien mit fester Länge](#)
- Tabellenbeziehungen und Filter: [Datenbankbeziehungen und wie man sie beibehält oder ignoriert](#)
- Definieren von SQL-Where-Filtern: [SQL WHERE-Komponente / -Beziehung](#)
- Tipps zu Filterkomponenten: [Filterkomponenten - Tipps](#)

13.2 Filterkomponenten - Tipps

Die "filter"-Komponente ist vor allem bei Datenbankabfragen wichtig, da Sie damit effizient mit großen Datenmengen arbeiten können. Bei Datenbanktabellen mit Tausenden von Zeilen können Sie mit Filtern den Zugriff auf die Tabelle einschränken und die Art, wie Daten extrahiert werden, effizient strukturieren. Die Art wie Filter verwendet werden, wirkt sich direkt auf die Geschwindigkeit der Mapping-Generierung aus.

In diesem Abschnitt werden Methoden erläutert, mit denen Sie den Zugriff auf Daten optimieren und den Mapping-Prozess insgesamt beschleunigen können.

In der Regel gilt: Verwenden Sie so wenige Filterkomponenten wie möglich und:

1. Vermeiden Sie das Verketteten von Filterkomponenten.
2. Verbinden Sie die "on-true/on-false"-Parameter, wenn möglich mit dem Parent-Element anstelle von Child-Elementen.
3. Verbinden Sie den "on-false"-Parameter, um das vom on-true-Parameter gelieferte Komplement-Nodeset zu mappen.
4. Verwenden Sie keine Filter zum Mappen von Child-Daten, wenn das Parent-Element gemappt wurde.
5. Verwenden Sie den "Prioritätskontext", um festzulegen in welcher Reihenfolge nicht miteinander in Beziehung stehende Elemente abgearbeitet werden sollen.

Vermeiden Sie das Verketteten von Filterkomponenten

Jede Filterkomponente führt dazu, dass die Quelldaten in einer Schleife verarbeitet werden, sodass **n** mal auf die Quelle zugegriffen wird. Wenn Sie zwei Filter miteinander verketteten, wird die Schleife **n*n** mal verarbeitet.

Lösung:

Verwenden Sie **"logical-and"** Komponenten, um die Booleschen Ausdrücke zweier Filterkomponenten miteinander zu kombinieren. Das Ergebnis ist eine einzige Filterkomponente, die die Schleife nur **n** mal abarbeitet.

Verbinden Sie den "on-true/on-false"-Parameter der Filterkomponente mit den Parent-Datenelementen des Ziels

Am besten funktionieren Filterkomponenten, wenn Sie, anstatt mit einzelnen Datenelementen, mit den entsprechenden Parent-Elementen verbunden werden.

Der Boolesche Filterausdruck wird daher gegen den Parent evaluiert, **bevor** die einzelnen Child-Elemente verarbeitet werden. Bei Verwendung von Filtern, die von einer Datenbanktabelle gemappt werden, wird folgende SQL-Anweisung generiert:

- "SELECT * FROM table WHERE <expression>" wenn das **Parent**-Datenelement gemappt wird oder
- "SELECT * FROM table" und anschließend für jede Zeile überprüfen, wenn die Child-Datenelemente gemappt werden.

Bitte beachten Sie:

Bei Verbindung eines Filters von einem Parent-Datenelement der Quelle muss der on-true/on-false-Parameter auch mit dem Parent-Datenelement des Ziels verbunden werden. Ist dies nicht möglich, wenden Sie diese Regel nicht an.

Verbinden Sie den "on-false"-Parameter, um das Komplement-Nodeset zu mappen

Indem Sie diesen Parameter verbinden, haben Sie schnellen Zugriff auf das durch das aktuelle Mapping definierte Komplement-Nodeset. Wenn Sie diesen Parameter verwenden, eine Verbindung mit Parent-Datenelementen herstellen usw., gelten dieselben Tipps.

Verwenden Sie keine Filter zum Mappen von Child-Daten, wenn das Parent-Element

gemappt wird

Bei Verwendung eines Filters zum Mappen von Daten von einem Parent der Quelle auf ein Parent des Ziels wird **derselbe Filter** automatisch auch auf **jedes Child**-Element dieses Parent-Elements angewendet.

Wenn das Parent-Datenelement gemappt werden kann, ist es nicht notwendig, Filter zu verwenden, um gefilterte Daten für die Child-Elemente zu liefern! Sie können daher Child-Daten direkt mappen.

Verwenden Sie den Prioritätskontext, um festzulegen, in welcher Reihenfolge nicht miteinander in Beziehung stehende Elemente gemappt werden sollen

Mappings werden immer von oben nach unten durchgeführt; wenn Sie zwei Tabellen durchsuchen/abarbeiten, wird zuerst eine Tabelle, dann die andere verarbeitet. Wenn Sie nicht miteinander in Beziehung stehende Elemente mappen, ohne den Prioritätskontext festzulegen, weiß MapForce nicht, welche Schleife zuerst verarbeitet werden soll, und wählt daher automatisch die erste Tabelle bzw. Datenquelle aus.

Lösung:

Legen Sie fest, welche Tabelle bzw. welche Quelldaten zuerst abgearbeitet/durchsucht werden sollen und legen Sie am Konnektor zu dieser Tabelle den Prioritätskontext fest. Ein konkreteres Beispiel finden Sie unter "[Prioritätskontext](#)".

So definieren Sie einen Prioritätskontext:

- Rechtsklicken Sie auf ein Input-Symbol und wählen Sie im Popup-Menü "Priorität - Kontext".
Steht die Option nicht zur Verfügung, müssen Sie zuerst die übrigen Input-Symbole mappen, damit die Option aktiv wird.

Filter und quellorientiertes (Mixed Content) Mapping

Quellorientierte Mappings funktionieren nur bei direkten Verbindungen zwischen Quell- und Zielkomponenten. Verbindungen unterhalb einer quellorientierten Verbindung werden nicht als quellorientiert betrachtet und die Datenelemente werden in der Reihenfolge Zielkomponentendatenelement/Node behandelt.

Ein einziger Filter, bei dem beide Output-Elemente mit demselben / separaten Zielelementen verbunden sind, verhält sich, als gäbe es zwei separate Filterkomponenten, wobei eine eine negierte Bedingung haben muss.

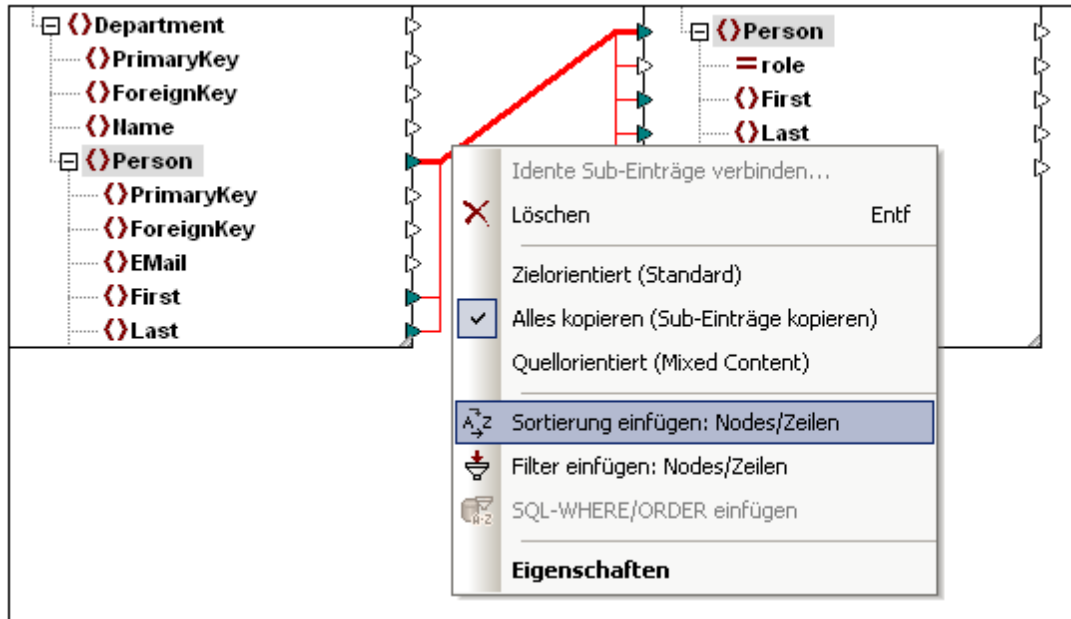
Wenn eine Ausnahmeereigniskomponente mit einer der Filterausgaben verbunden ist, so wird die Ausnahmebedingung überprüft, wenn die Mappings auf die andere Filterausgabe ausgeführt werden.

13.3 Sortierkomponente - Sortieren von Input-Sequenzen

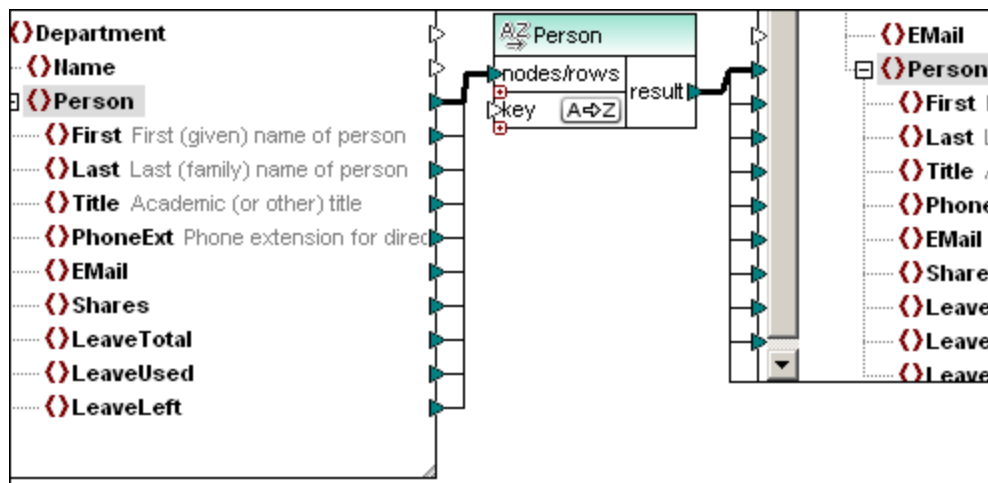
Um Input-Daten auf Basis eines bestimmten Sortierschlüssels zu sortieren, verwenden Sie bitte die **Sortierkomponente** ("sort"). Die Sortierkomponente unterstützt derzeit die Ziele XSLT2, XQuery und den Built-in-Ausführungsprozessor.

So fügen Sie eine Sortierkomponente ein:

1. Klicken Sie mit der rechten Maustaste auf einen Konnektor zwischen den zu sortierenden Nodes.



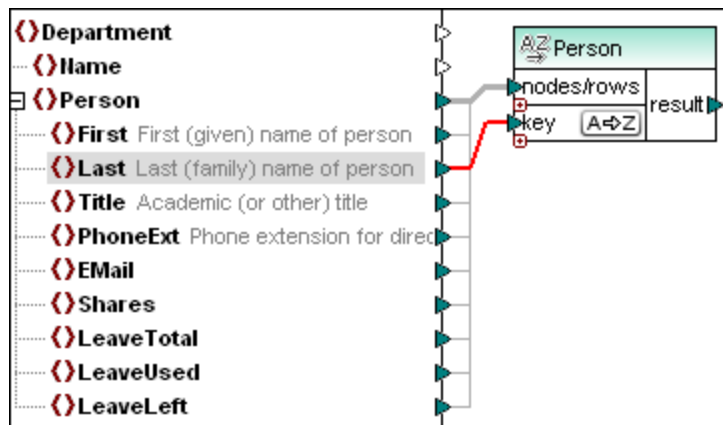
2. Wählen Sie im Kontextmenü den Befehl **Sortierung einfügen: Nodes/Zeilen**.



Daraufhin wird eine Sortierkomponente eingefügt und automatisch mit der Quell- und der Zielkomponente verbunden.

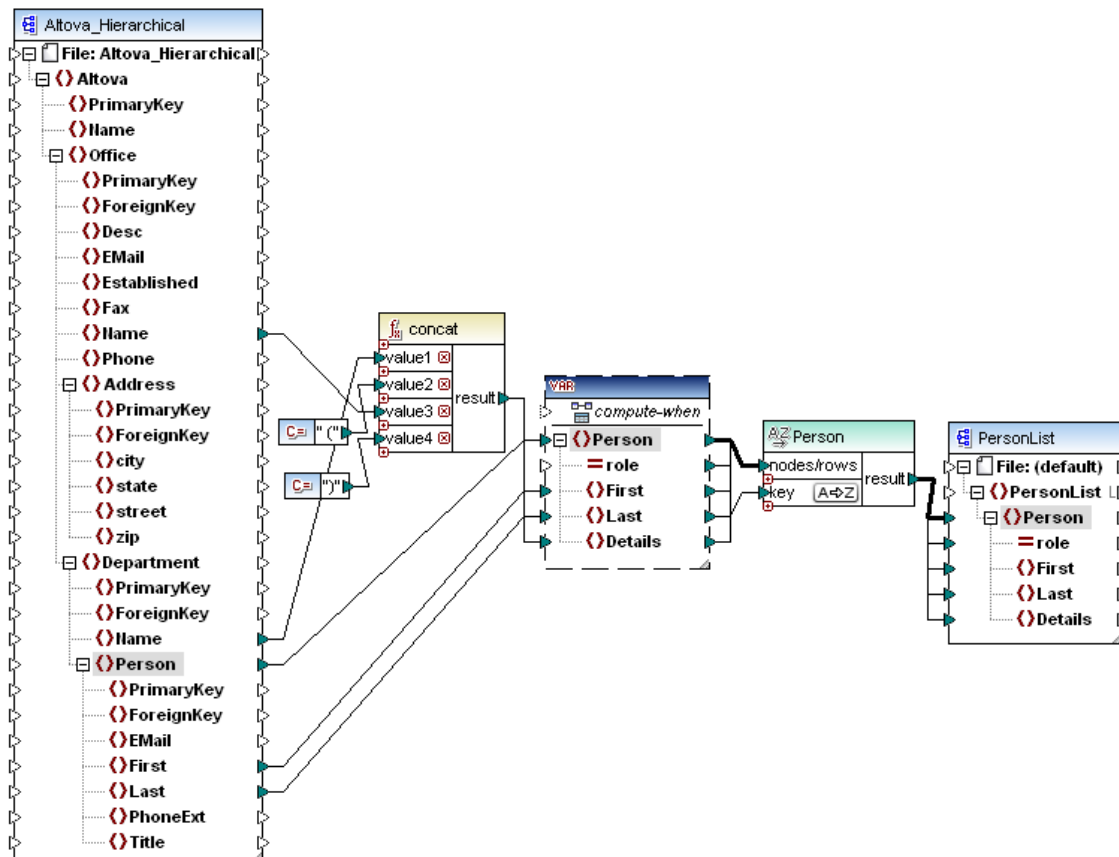
So definieren Sie, nach welchem Datenelement sortiert werden soll:

- Verbinden Sie das Datenelement, nach dem sortiert werden soll, z.B. **Last**, mit dem **key**-Parameter der Sortierkomponente, die nun den Namen "Person" hat.



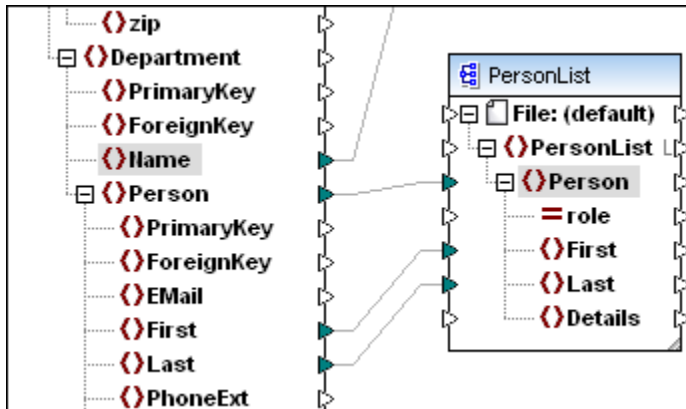
Die Personen werden nun auf dem Ausgaberegister nach ihrem Nachnamen (Last) geordnet.

Beispiel: **Altova_Hierarchical_Sort.mfd** im Ordner MapForceExamples.



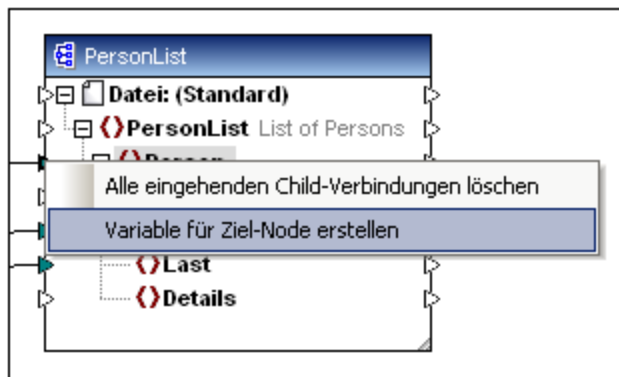
Ziel ist es, die Personen der einzelnen Niederlassungen (branch office) alphabetisch zu sortieren und detaillierte Informationen zur Niederlassung (office) und zur Abteilung (department) zu inkludieren. In diesem Beispiel haben wir uns der Komponente "Variable" bedient, um Zugriff auf übergeordnete Datenelemente zu erhalten, die sonst nicht zur Verfügung stünden. In diesem Fall handelt es sich um übergeordnete Datenelemente des Node "Person".

Schritte zur Erstellung dieses Mappings:
Unten sehen Sie die erste Phase des Mappings.

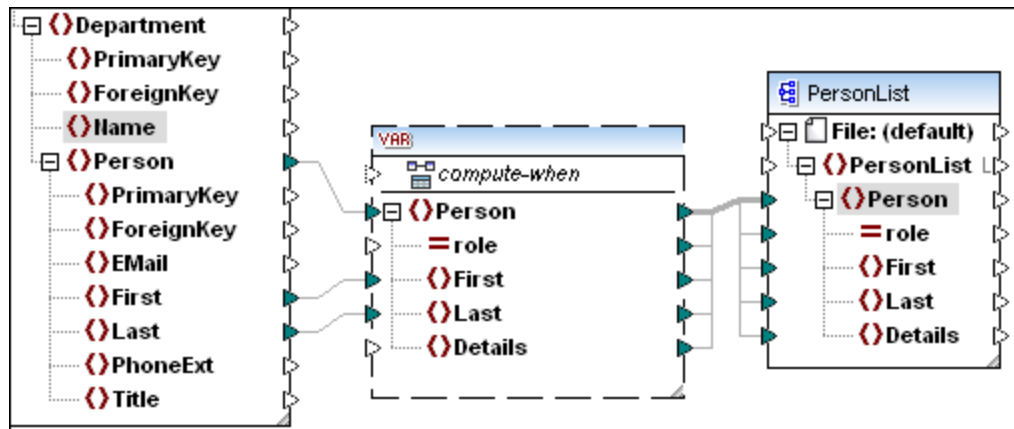


Da die Sortierung nach einem Person-Datenelement vorgenommen werden soll, werden die Person-Datenelemente in beiden Komponenten miteinander verbunden, wobei auch untergeordnete Datenelemente mit identischen Namen miteinander verbunden werden.

1. Klicken Sie mit der rechten Maustaste auf das Input-Symbol (genau auf das Dreieck) neben dem Datenelement "Person" der Zielkomponente.
2. Wählen Sie im Kontextmenü den Eintrag "Variable für Ziel-Node erstellen".



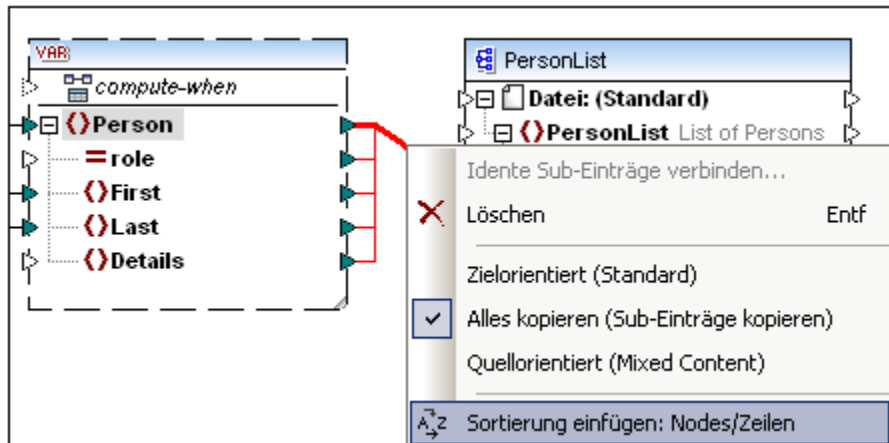
Daraufhin wird eine komplexe Zwischenvariable zwischen den Komponenten eingefügt, wobei gleichnamige Datenelemente miteinander verbunden werden. Nähere Informationen dazu finden Sie unter [Zwischenvariablen](#).



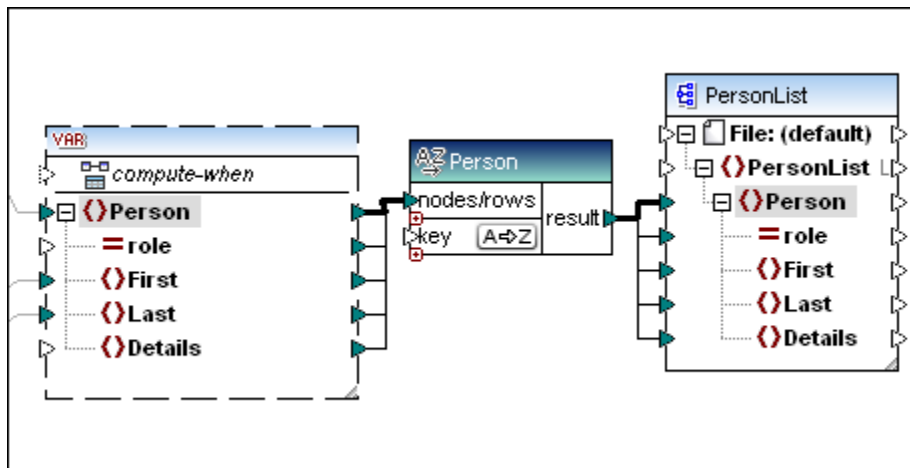
Da die Personen nach dem Nachnamen (Last) sortiert werden sollen, müssen wir nun

die Sortierkomponente einfügen.

3. Klicken Sie mit der rechten Maustaste auf den "Alles kopieren"-Konnektor und wählen Sie den Befehl "Sortierung einfügen: Nodes/Zeilen".

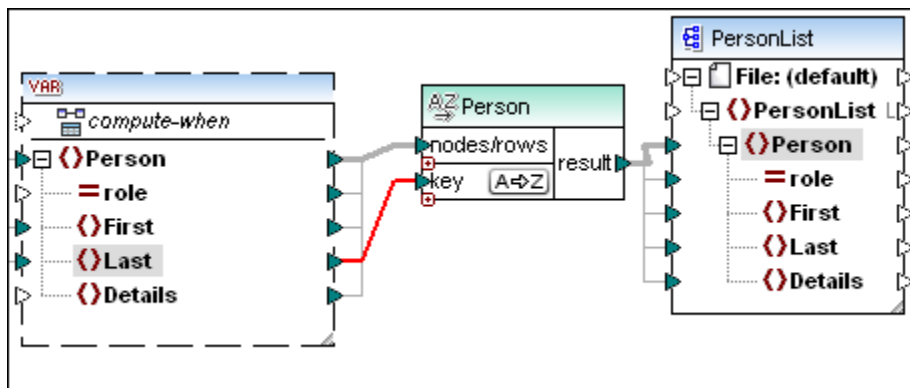


Daraufhin wird die Sortierkomponente eingefügt und alle relevanten Datenelemente werden automatisch verbunden.



Nun muss nur noch der Schlüssel definiert werden, nach dem sortiert werden soll.

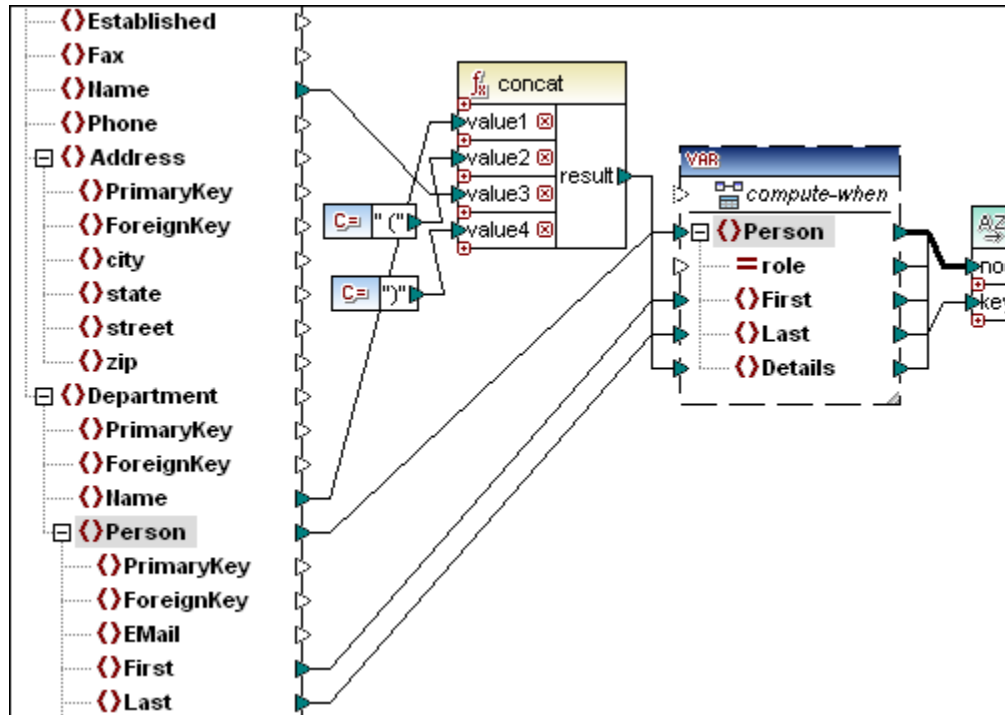
4. Verbinden Sie das Datenelement "Last" der Zwischenvariable mit dem **key**-Parameter der Sortierkomponente.



5. Klicken Sie auf die Schaltfläche "Ausgabe" um eine Vorschau des Ergebnisses zu sehen.

Die Personen sind nun nach ihrem Nachnamen sortiert. Klicken Sie auf die Schaltfläche "Mapping" um zum Design-Fenster zurückzuwechseln.

6. Verbinden Sie nun die restlichen Datenelemente wie unten gezeigt mit Hilfe der concat-Funktion und der Konstantenkomponenten (für die Klammern).
7. Verbinden Sie den result-Parameter der concat-Funktion mit dem Datenelement "Details" der Zwischenvariable.



7. Klicken Sie auf die Schaltfläche "Ausgabe" um das Ergebnis zu sehen.



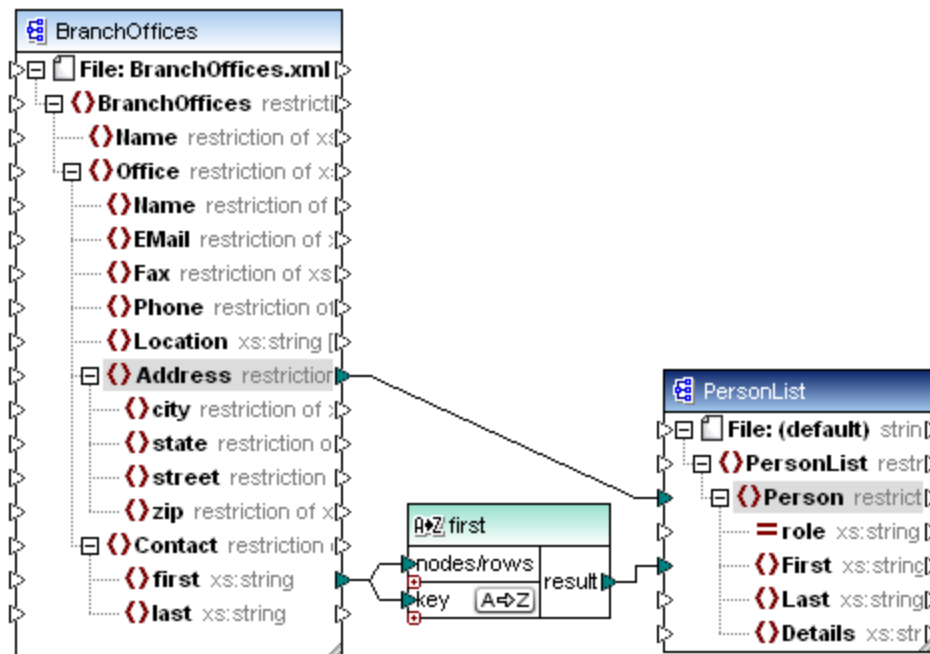
Die Personen wurden wie zuvor nach ihrem Nachnamen sortiert, doch wurden Zusatzinformationen aus dem Feld "Details" zu den einzelnen Personendatensätzen hinzugefügt. Für jede Person steht nun der Name der richtigen Niederlassung (office) und Abteilung (department) zur Verfügung, da über die Zwischenvariable Daten aus übergeordneten Nodes von einem untergeordneten Node aus abgerufen werden können. Dies kann nur mit Hilfe der Zwischenvariablen bewerkstelligt werden.

So kehren Sie die Sortierreihenfolge um:

- Klicken Sie in der Sortierkomponente auf die Schaltfläche **A↔Z**. Daraufhin ändert diese sich in **Z↔A**, um anzuzeigen, dass die Reihenfolge umgekehrt wurde.

So sortieren Sie Input-Daten, die aus Einträgen vom Typ "simpleType" bestehen:

- Verbinden Sie den simpleType-Eintrag, z.B. first, **sowohl** mit dem nodes/row- **als auch** dem key-Parameter der Sortierkomponente.

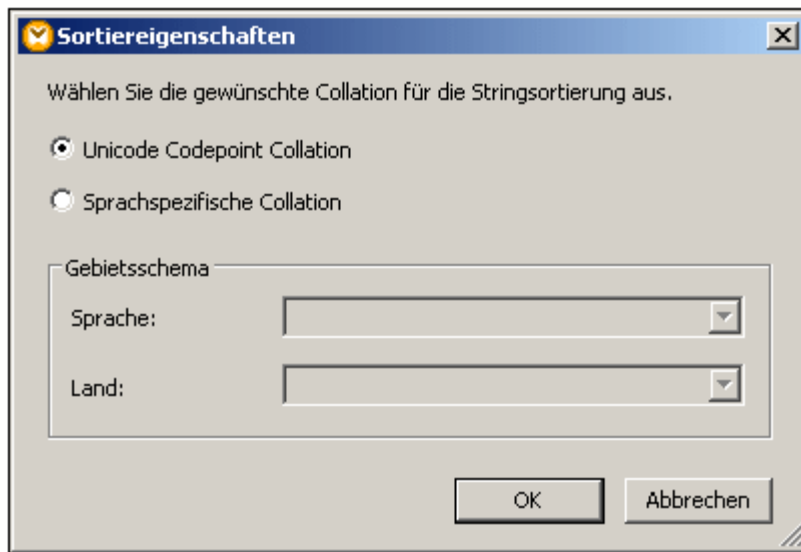


Sortieren von Datenbankdaten - Ziel: Built-in (Ausführungsprozessor)

Mit Hilfe der Sortierkomponente können auch Tabellendaten sortiert werden. Besser funktioniert dies jedoch mit Hilfe der [SQL-WHERE/ORDER](#)-Komponente.

So sortieren Sie Strings mit Hilfe von sprachspezifischen Regeln:

Doppelklicken Sie auf die Titelleiste/Überschrift der eingefügten Sortierkomponente um das Dialogfeld "Sortiereigenschaften" zu öffnen.



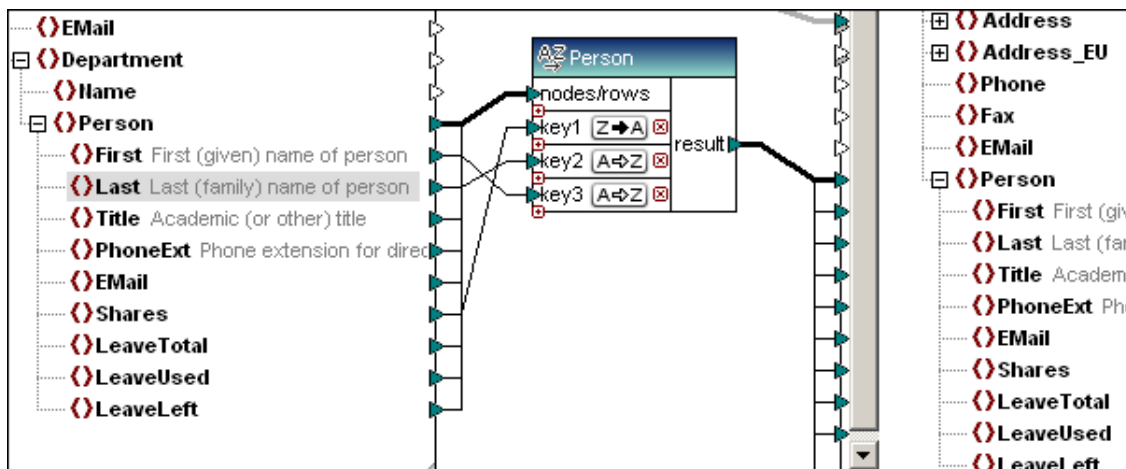
Unicode Codepoint Collation: Mit dieser Standardoption werden Strings auf Basis ihrer Codepoint-Werte verglichen/sortiert. Codepoint-Werte sind Ganzzahlen, die abstrakten Zeichen in dem vom Unicode Consortium abgesegneten universalen Zeichensatz zugewiesen sind. Mit Hilfe dieser Option kann eine Sortierung in den verschiedensten Sprachen und Scripts vorgenommen werden.

Sprachspezifische Collation: Mit dieser Option können Sie die Variante für eine bestimmte Sprache und ein bestimmtes Land festlegen, nach dem sortiert werden soll. Diese Option wird bei Verwendung des BUILTIN-Ausführungsprozessors und - je nachdem welcher Prozessor für die Ausführung des Codes verwendet wird - für XSLT unterstützt.

So sortieren Sie nach mehreren Schlüsseln:

Die Sortierkomponente unterstützt auch das Definieren mehrerer Sortierschlüssel.

- Wenn Sie auf das "+" Symbol klicken wird ein neuer Schlüssel zur Sortierkomponente hinzugefügt, in diesem Fall key2
- Wenn Sie auf das "x" Symbol klicken, wird der entsprechende Schlüssel gelöscht.
- Wenn Sie mit der Maus einen Konnektor auf ein + Symbol ziehen, wird der Parameter automatisch eingefügt/hinzugefügt und eine Verbindung dazu hergestellt.

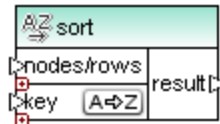


Beachten Sie dass key1 eine höhere Sortierpriorität als key2 und key2 eine höhere als key3 hat.

So fügen Sie eine Sortierkomponente auf konventionelle Weise ein:

- Klicken Sie in der Symbolleiste auf die Schaltfläche "Sortierkomponente einfügen"  um die Komponente einzufügen.

Daraufhin wird die Sortierkomponente in nicht verbundener Form, in der "sort" in der Titelleiste der Komponente zu sehen ist, eingefügt.



Sobald eine Verbindung zur Quellkomponente erstellt wurde, ändert sich der Name in der Titelleiste in den des mit dem Parameter nodes/rows verbundenen Eintrags.

13.4 Wertezuordnung - Transformieren von Input-Daten

Mit Hilfe der Wertezuordnungskomponente können Sie einen Input-Wert mit Hilfe einer Lookup-Tabelle in einen anderen Output-Wert transformieren. Dies eignet sich zum Konvertieren unterschiedlicher Enumerationstypen. Die Komponente hat nur ein Input- und ein Output-Datenelement.

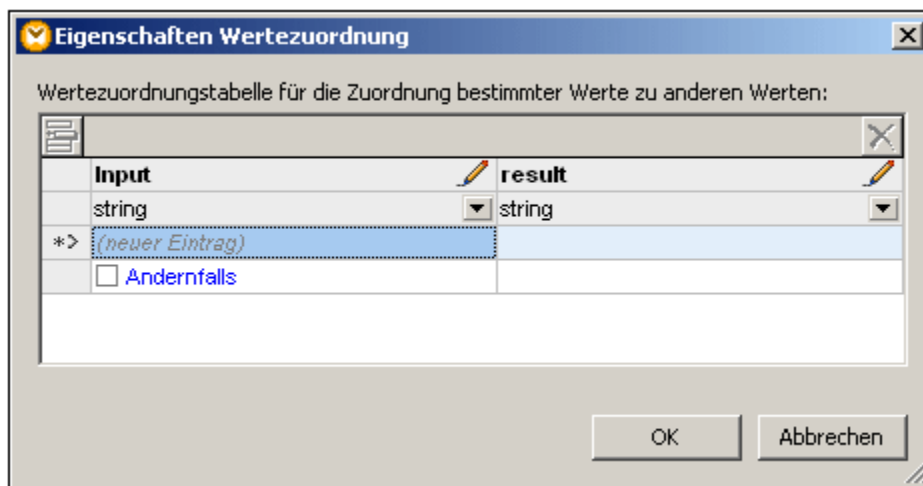
Anmerkung: Wenn Sie Daten anhand von bestimmten Kriterien abrufen/filtern möchten, verwenden Sie bitte die **Filter**-Komponente (siehe [Lookup-Tabelle, - Abrufen dynamischer Daten](#) im folgenden Abschnitt, Filtern von XML-Daten: [Filtern von Daten](#), Filtern von CSV-Dateien: [Hierarchien für CSV-Dateien und Dateien mit fester Länge](#)).

So verwenden Sie eine Wertezuordnungskomponente:

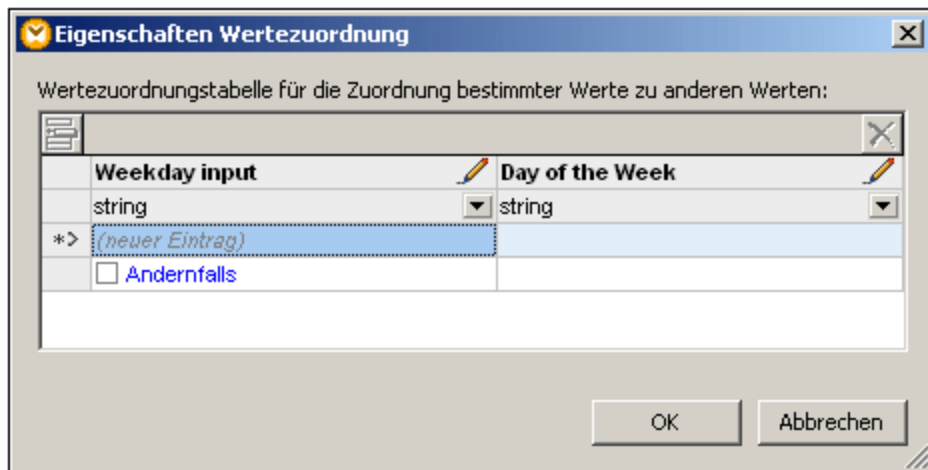
1. Wählen Sie die Menüoption **Einfügen | Wertezuordnung** oder klicken Sie in der Symbolleiste auf die Schaltfläche "Wertezuordnung" .



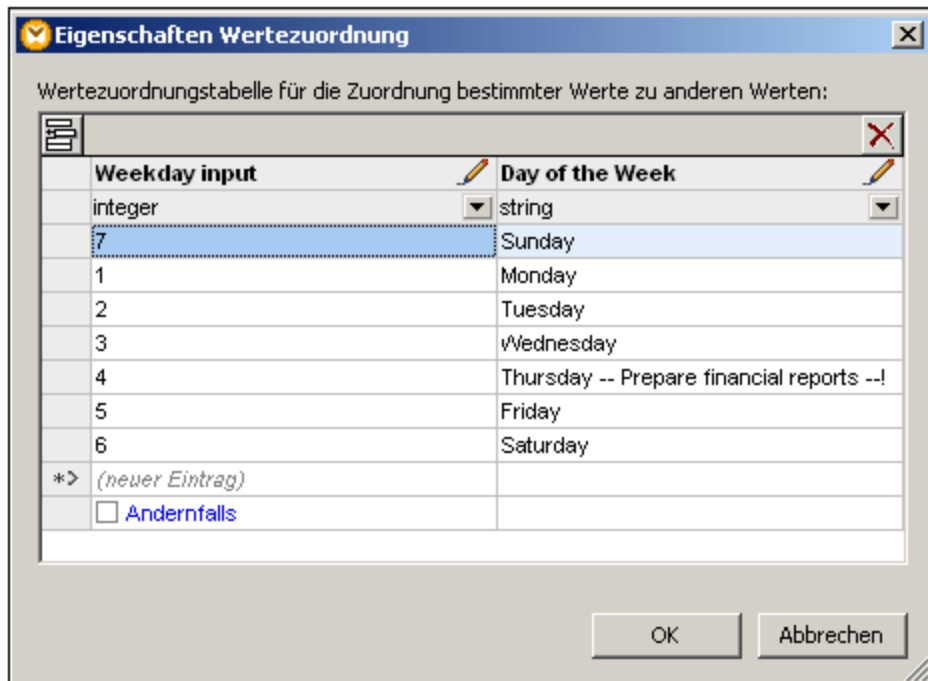
2. Doppelklicken Sie auf die Wertezuordnungskomponente, um die Wertezuordnungstabelle zu öffnen.



3. Klicken Sie in die Spaltenüberschriften und geben Sie in die erste Spalte **Weekday input** und in die zweite **Day of the Week** ein.

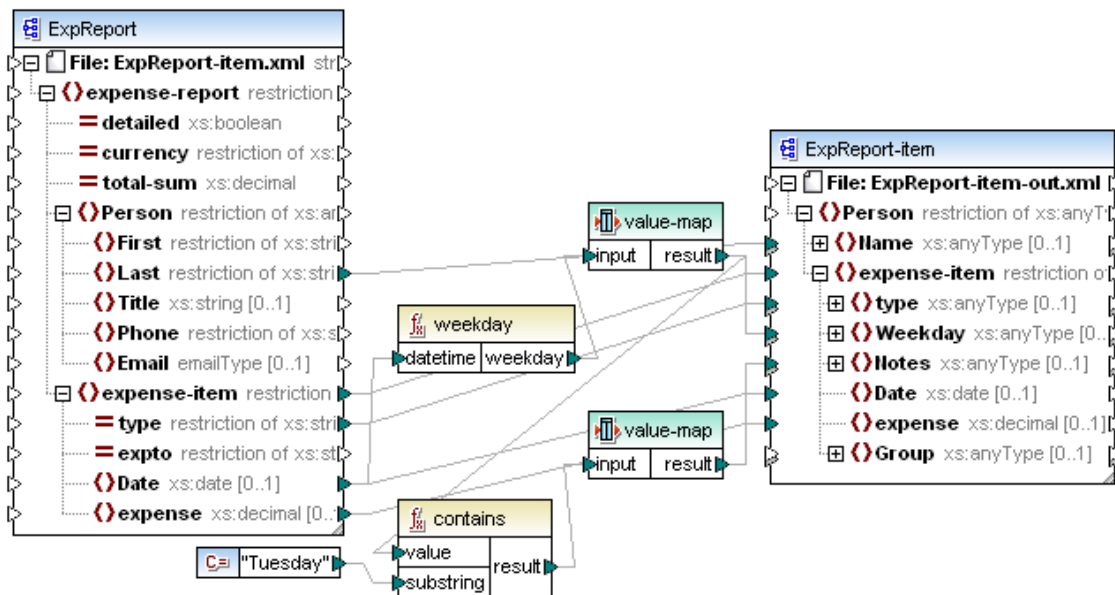


4. Geben Sie den Input-Wert, der transformiert werden soll, in die Spalte **Weekday input** ein.
5. Geben Sie den Ausgabewert, in den der Wert transformiert werden soll, in die Spalte **Day of the week** ein.
6. Geben Sie ein neues Wertepaar in das Input-Feld **(neuer Eintrag)** ein.
6. Klicken Sie auf die **Datentyp**-Auswahlliste unterhalb der Spaltenüberschrift, um den Input- und den Output-Datentyp auszuwählen, z.B.: Integer und String.



7. Anmerkung: Aktivieren Sie das Kontrollkästchen **Andernfalls** und geben Sie den Wert ein, um einen alternativen Ausgabewert zu definieren, wenn die bereitgestellten Werte in der Input-Komponente nicht vorhanden sind. Nähere Informationen dazu, wie Sie Quelldaten übergeben, ohne diese zu ändern finden Sie unter [Übergeben von unveränderten Daten über eine Wertezuordnung](#).
8. Um die Spaltennamen, die auch im Mapping angezeigt werden, zu ändern, klicken Sie auf die Bearbeitungssymbole in den Kopfzeilen. Auf diese Art kann der Zweck der Komponente im Mapping besser verdeutlicht werden.

Die Datei **Expense-valmap.mfd** im Ordner **...MapForceExamples\Tutorial** ist ein Beispiel-Mapping, in dem Sie sehen, wie die Wertezuordnung verwendet werden kann.



Funktionsweise dieses Mappings:

Der Wochentag wird aus dem Datenelement "Date" in der Datenquelle extrahiert, der numerische Wert wird in Text konvertiert und dieser Text (also Sunday, Monday, usw.) wird in das Datenelement "Weekday" der Zielkomponente eingesetzt.

- Die Funktion **weekday** extrahiert die Zahl für den Wochentag aus dem Datenelement **Date** in der Quelldatei "ExpReport". Das Ergebnis dieser Funktion sind Ganzzahlen von 1 bis 7.
- Die Wertezuordnungskomponente transformiert die Ganzzahlen in Wochentage, also i. e. Sunday, Monday, usw., wie in der Grafik oben in diesem Abschnitt gezeigt.
- Wenn die Ausgabe den Wert "Tuesday" enthält, so wird die entsprechende Ausgabe "Prepare Financial Reports" auf das Datenelement "Notes" in der Zielkomponente gemappt.
- Wenn Sie auf die Schaltfläche "Ausgabe" klicken, wird die XML-Zieldatei mit den transformierten Daten angezeigt.

```

3      <Name>Landis</Name>
4      <expense-item>
5          <type>Meal</type>
6          <Weekday>Tuesday</Weekday>
7          <Notes>-- Prepare financial reports -- !</Notes>
8          <Date>2003-01-01</Date>
9          <expense>122.11</expense>
10     </expense-item>
11     <expense-item>
12         <type>Lodging</type>
13         <Weekday>Monday</Weekday>
14         <Notes> --</Notes>
15         <Date>2003-01-14</Date>
16         <expense>122.12</expense>
17     </expense-item>

```

Anmerkung:

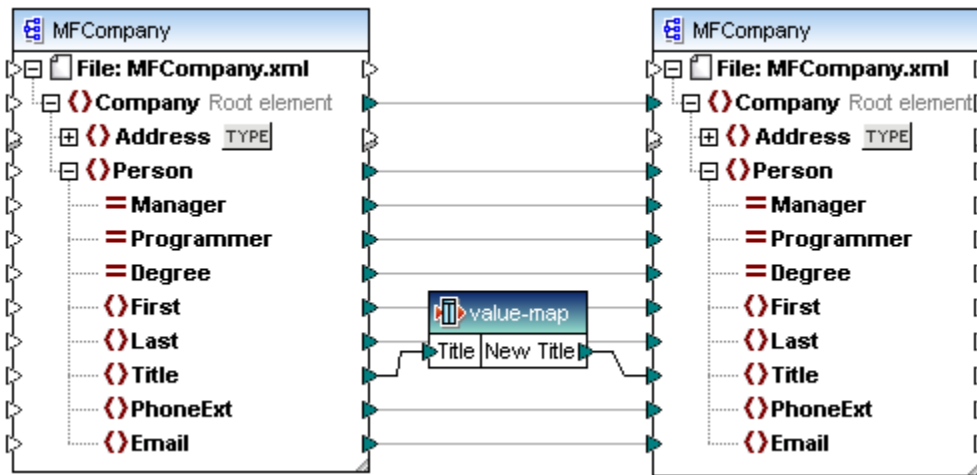
Wenn Sie den Mauszeiger über die Wertezuordnungskomponente platzieren, wird ein Popup-Fenster geöffnet, das die aktuell definierten Werte enthält.

Die Ausgabe von verschiedenen Typen von logischen Funktionen oder String-Funktionen kann nur einen Bool'schen Wert "**true**" oder "**false**" haben. Der Wert, den Sie überprüfen möchten, muss daher in das **Input**-Feld der Wertezuordnungstabelle eingegeben werden, z.B. "true".

13.4.1 Übergeben von unveränderten Daten über eine Wertezuordnung

In diesem Abschnitt wird eine Mapping-Situation beschrieben, in der bestimmte Node-Daten transformiert werden müssen, während die restlichen Node-Daten unverändert an den Ziel-Node übergeben werden müssen.

Ein Beispiel dafür wäre eine Firma, die einige der Berufsbezeichnungen in einer ihrer Niederlassungen ändert. In diesem Fall sollen zwei Berufsbezeichnungen (Title) geändert werden, während der Rest unverändert beibehalten werden soll.



Oben sehen Sie ein Mapping dafür, in dem, wie erwartet, zum Transformieren der betreffenden Bezeichnungen eine Wertezuordnungskomponente verwendet wird.

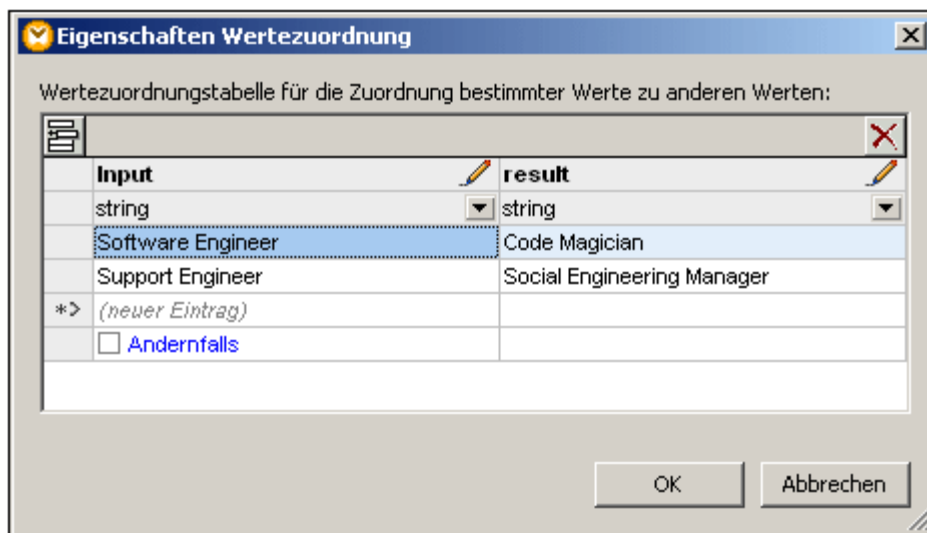
Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie das Ergebnis des Mappings.

```

33  <Person>
34    <First>Fred</First>
35    <Last>Landis</Last>
36    <PhoneExt>951</PhoneExt>
37    <Email>f.landis@nanonull.com</Email>
38  </Person>
39  <Person>
40    <First>Michelle</First>
41    <Last>Butler</Last>
42    <Title>Code Magician</Title>
43    <PhoneExt>654</PhoneExt>
44    <Email>m.landis@nanonull.com</Email>
45  </Person>

```

Bei den Personen, deren Berufsbezeichnung (Title) mit keinem der beiden in der Wertezuordnungskomponente angezeigten Typen übereinstimmt, wird das Element "Title" in der Ausgabedatei gelöscht.



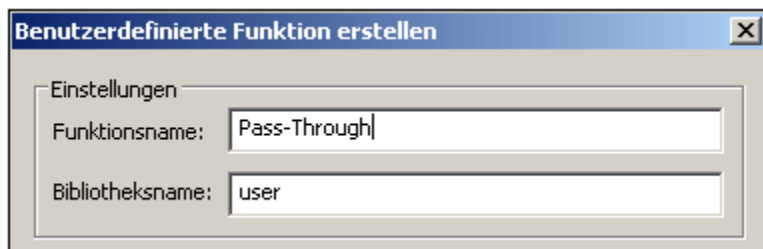
Mögliche Alternative:

Wenn Sie das Kontrollkästchen **Andernfalls** aktivieren und eine Ersatzbezeichnung eingeben, wird die Berufsbezeichnung in der Ausgabedatei zwar wieder angezeigt, enthält aber bei allen Personen in der Firma nun dieselbe **neue Berufsbezeichnung**.

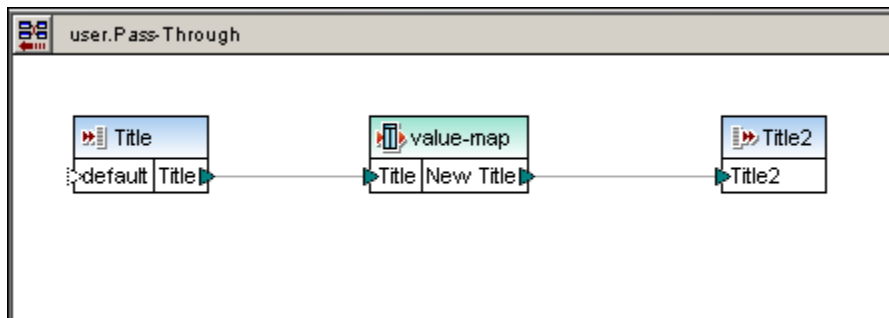
Lösung:

Erstellen Sie eine benutzerdefinierte Funktion, die die Wertezuordnungs-komponente enthält und verwenden Sie die **substitute-missing**-Funktion, um an leere Nodes die Originaldaten zu übergeben.

1. Klicken Sie auf die value-map-Komponente und wählen Sie den Befehl **Funktion | Benutzerdefinierte Funktion von Auswahl erstellen**.

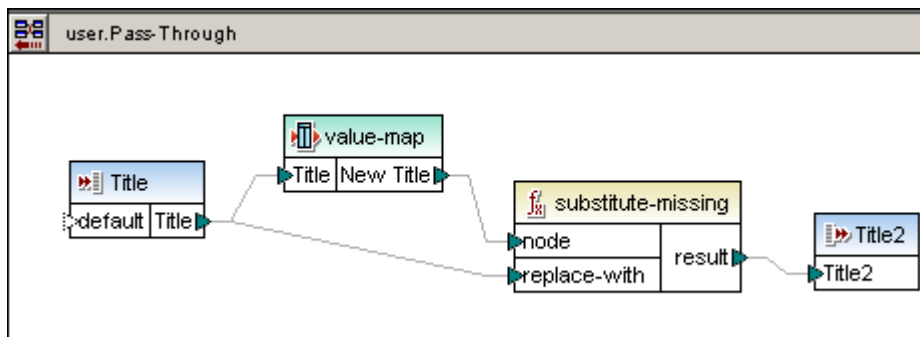


2. Geben Sie einen Namen für die Funktion ein, z.B. Pass-Through und klicken Sie auf OK.



3. Fügen Sie eine **substitute-missing**-Funktion aus dem Abschnitt **core | node function** des Bibliotheks-fensters ein und stellen Sie, wie in der Abbildung unten gezeigt, die

Verbindungen her.



4. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis zu sehen:

Mapping-Ergebnis:

- Die beiden Berufsbezeichnungen in "Title" wurden in "New Title" transformiert.
- Alle anderen Title-Nodes der Quelldatei behalten ihre ursprünglichen Title-Daten in der Zieldatei bei.

```

38  <Person>
39    <First>Fred</First>
40    <Last>Landis</Last>
41    <Title>Program Manager</Title>
42    <PhoneExt>951</PhoneExt>
43    <Email>f.landis@nanonull.com</Email>
44  </Person>
45  <Person>
46    <First>Michelle</First>
47    <Last>Butler</Last>
48    <Title>Code Magician</Title>
49    <PhoneExt>654</PhoneExt>
50    <Email>m.landis@nanonull.com</Email>
51  </Person>

```

Warum geschieht das?

Die Wertezuordnungskomponente wertet die Input-Daten aus.

- Wenn die eingehenden Daten mit einem der Einträge in der ersten Spalte **übereinstimmen**, werden die Daten transformiert und an den Node-Parameter von substitute-missing und dann weiter an Title2 übergeben.
- Wenn die eingehenden Daten **mit keinem der Einträge in der linken Spalte übereinstimmen**, wird kein Wert von der Wertezuordnung an den Node-Parameter übergeben, d.h. dies ist ein **leerer Node**.

Wenn das passiert, ruft die substitute-missing-Funktion den Original-Node und die Daten aus dem Title-Node ab und übergibt diese über den **replace-with** Parameter an Title2.

13.4.2 Eigenschaften der Wertezuordnungskomponente

Aktionen:



Klicken Sie auf die Schaltfläche "Einfügen", um vor der aktiven Zeile eine neue Zeile **einzu**fügen.



Klicken Sie auf die Schaltfläche "Löschen", um die aktive Zeile zu **lös**chen.

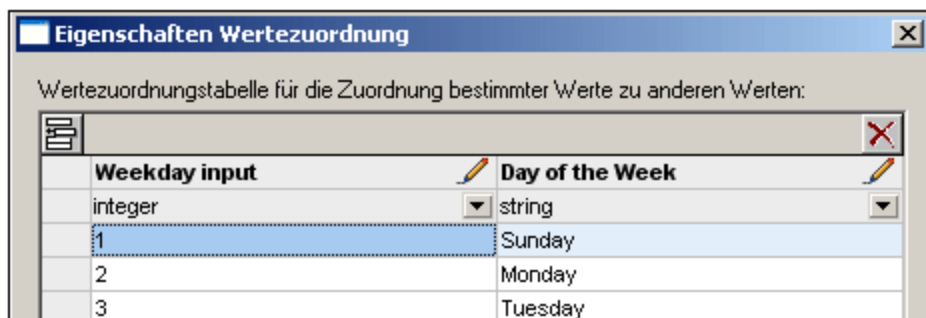


Klicken Sie auf die Schaltfläche "Bearbeiten", um die Spaltenüberschrift zu **be**arbeiten.

Sie können die Reihenfolge von Zeilen auch durch Ziehen mit der Maus ändern.

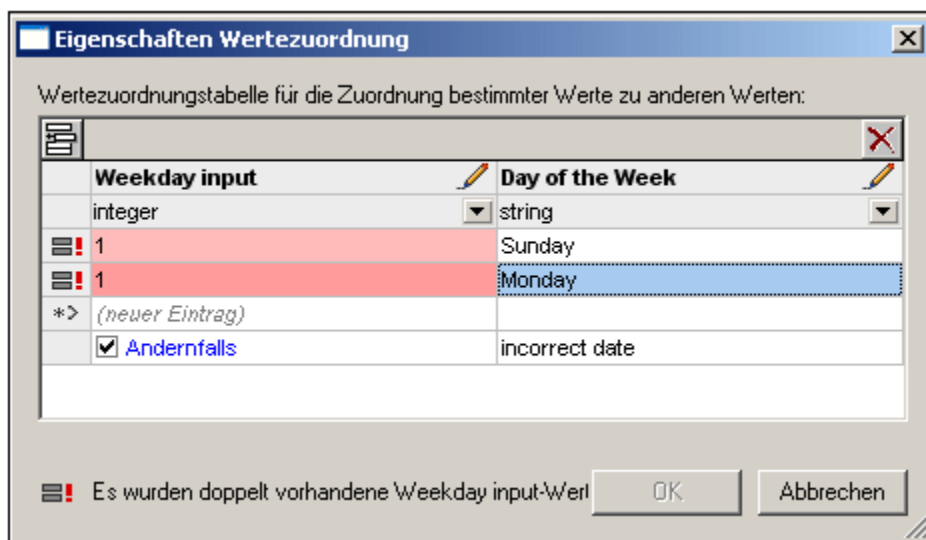
Ändern der Spaltenüberschrift:

Doppelklicken Sie auf die Spaltenüberschrift oder klicken Sie auf das Bleistiftsymbol, um den Spaltennamen zu bearbeiten und einen sinnvollen Namen einzugeben. Auf diese Art lässt sich der Zweck der Komponente besser erkennen, da die Spaltennamen auch im Mapping angezeigt werden.



Verwendung eindeutiger Input-Werte:

Die in die Spalte "Input" eingegebenen Werte müssen eindeutig sein. Wenn Sie zwei identische Werte eingeben, werden beide automatisch markiert, damit Sie einen davon korrigieren können.

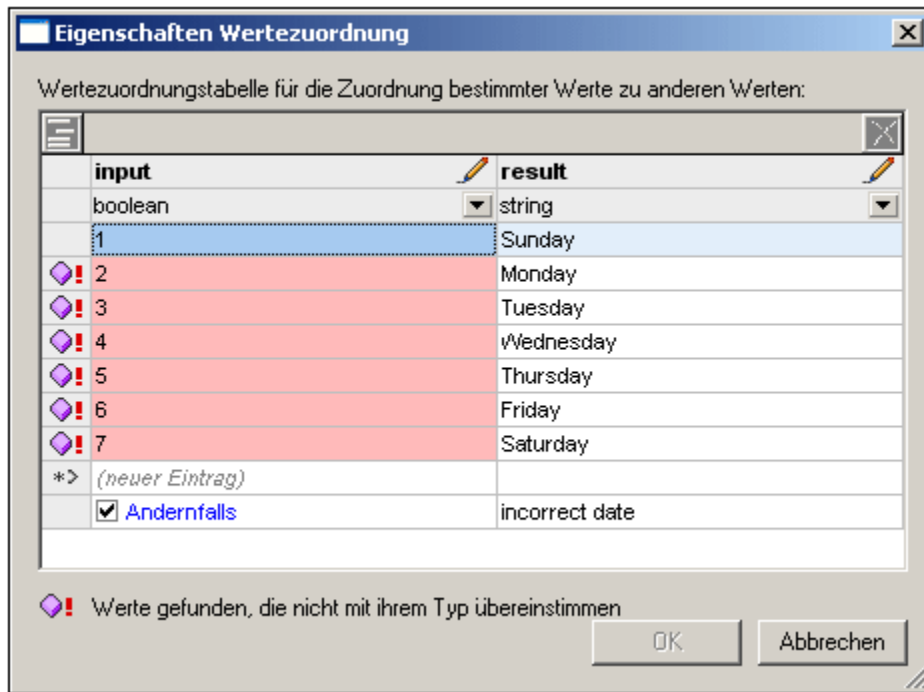


Sobald Sie einen der Werte korrigiert haben, ist die OK-Schaltfläche wieder aktiv.

Input- und Output-Datentypen

Die Input- und Ergebnis-Datentypen werden automatisch überprüft, wenn Sie eine Auswahl über die Auswahlliste treffen. Wenn ein Wert nicht übereinstimmt, werden die entsprechenden Felder markiert und die Schaltfläche "OK" wird deaktiviert. Ändern Sie den Datentyp in einen unterstützten Datentyp.

In der Abbildung unten wurden ein Boolescher Wert und ein String ausgewählt.



13.5 Ersetzen von Sonderzeichen in Datenbankdaten

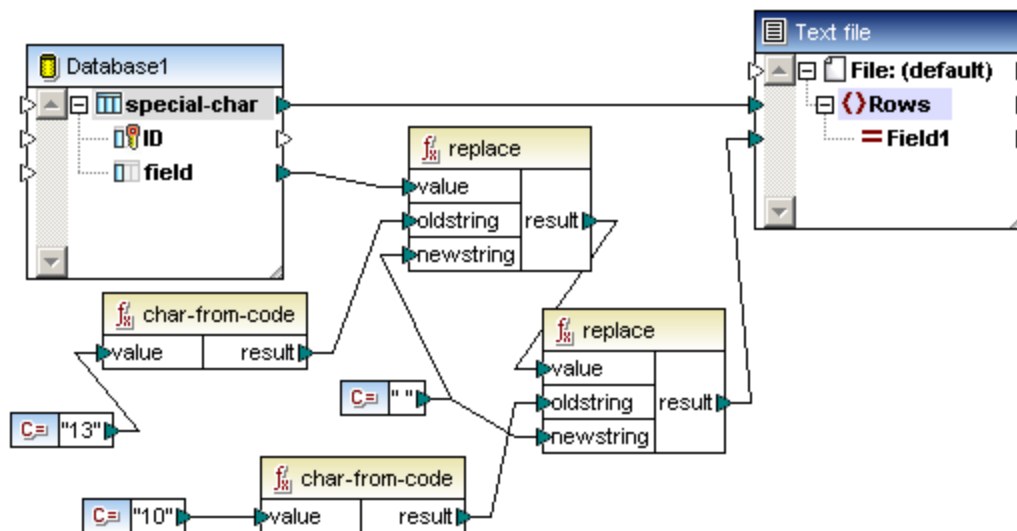
Beim Transformieren von Datenbankdaten müssen Sie unter Umständen bestimmte Sonderzeichen wie z.B. newline (Zeilenumbruchszeichen) CR/LF (hex 0D/0A) aus der Datenquelle entfernen. Dies können Sie mit Hilfe der "char-from-code"-Funktion aus der String-Funktionsbibliothek bewerkstelligen.

Im unten gezeigten Mapping ist die Datenquelle eine MS Access-Datenbank bestehend aus einer einzigen Tabelle mit zwei Zeilen. Der Text jeder der beiden Tabellenzeilen enthält mehrere Zeilen, die durch Zeilenumbrüche (LF) hex 0D, dezimal 13 getrennt sind.

Table1
A
This is
which will be

Im Mapping wird Folgendes durchgeführt:

- Die Zeichen 0D und 0A hex (dezimal 13 und 10) werden in einen String konvertiert, der von der Ersetzungsfunktion weiter verarbeitet werden kann.
- Mit Hilfe der Ersetzungsfunktion wird der Parameter "oldstring", der das Ergebnis der char-from-code (0D/0A) Funktion ist, durch das Leerzeichen ersetzt, das von der Konstantenkomponente als Parameter "newstring" bereitgestellt wird.
- Die konvertierten Daten werden in eine Textkomponente platziert.



Wenn Sie auf das Ausgaberegister klicken, wird das Ergebnis im Vorschaufenster angezeigt. Die Zeilen (CR) LF in den einzelnen Datenbankfeldern wurden durch ein Leerzeichen ersetzt.

1	This is our new company policy
2	which will be implemented immediately.
3	
4	
5	
6	

13.6 Aggregatfunktionen: min, max, sum, count, avg

Aggregatfunktionen dienen zum Ausführen von Operationen an einer Gruppe von Eingabewerten (oder an einer Folge von Werten) im Gegensatz zu einem einzigen Wert. Die Aggregatfunktionen befinden sich in der Core-Bibliothek. Das unten gezeigte Mapping finden Sie im Ordner ...\\Tutorial unter dem Namen **Aggregates.mfd**.

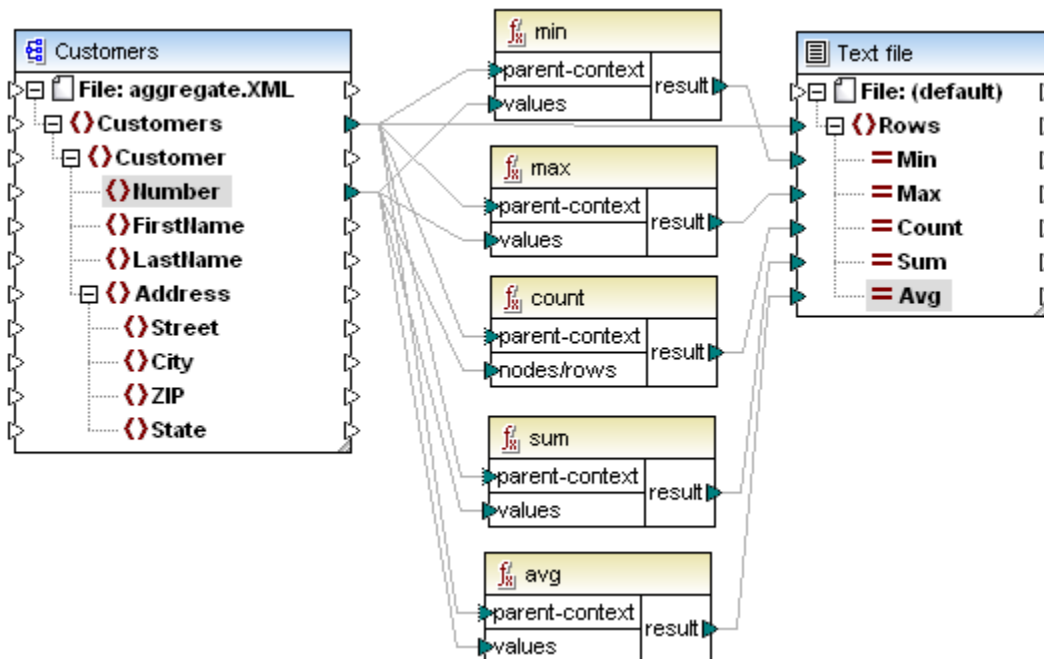
Informationen zum Erstellen von Aggregatfunktionen mit Hilfe von Named Templates finden Sie unter [Aggregatfunktionen - Aggregieren von Nodes in XSLT1 und 2](#).

Bibliotheken	
core	
aggregates	
avg	result = avg(values)
count	result = count(nodes/rows)
max	result = max(values)
min	result = min(values)
string-join	result = string-join(strin
sum	result = sum(values)

Java ausgewählt

Aggregatfunktionen haben zwei Input-Elemente.

- **values** (nodes/rows) ist mit dem Quellelement verbunden, aus dem die Daten stammen, in diesem Fall "Number".
- **parent-context** ist mit dem Element verbunden, an dem iteriert werden soll, in diesem Fall allen Kunden ("Customer"). Der Parameter ist allerdings optional.



Bei der Input-Instanz handelt es sich in diesem Fall um eine XML-Datei mit den folgenden Daten:

Comment edited with XMLSPY v2004 U (<http://www.xmlspy.com>) by

Customers																
xmlns:xsi	http://www.w3.org/2001/XMLSchema															
xsi:noNamespace...	Customers.xsd															
Customer (4)																
	<table> <tr> <th></th><th>Number</th><th>FirstName</th></tr> <tr> <td>1</td><td>2</td><td>FredJohn</td></tr> <tr> <td>2</td><td>4</td><td>MichelleAnn-marie</td></tr> <tr> <td>3</td><td>6</td><td>TedMac</td></tr> <tr> <td>4</td><td>8</td><td>AnnLong</td></tr> </table>		Number	FirstName	1	2	FredJohn	2	4	MichelleAnn-marie	3	6	TedMac	4	8	AnnLong
	Number	FirstName														
1	2	FredJohn														
2	4	MichelleAnn-marie														
3	6	TedMac														
4	8	AnnLong														

- Bei den Quelldaten, die an das Element "values" übergeben werden, handelt es sich um die Zahlenfolge 2,4,6,8.
- Bei der Output-Komponente handelt es sich in diesem Fall um eine einfache Textdatei. Wenn Sie für das obige Mapping auf das Register "Ausgabe" klicken, erhalten Sie das folgende Ergebnis:

1	2,8,4,20,5
2	

min=2, max=8, count=4, sum=20 and avg=5.

13.7 Mappen mehrerer Tabellen auf eine XML-Datei

Mappen mehrerer hierarchischer Tabellen auf eine XML-Ausgabedatei

- Sie haben eine Datenbank und möchten eine bestimmte Anzahl von Tabellen extrahieren und auf eine XML-Datei mappen.
- Zwischen den Tabellen bestehen Primär- und Sekundärschlüsselbeziehungen.
- Verwandte Tabellen sollen in der erzeugten XML-Datei als Child-Elemente angezeigt werden.

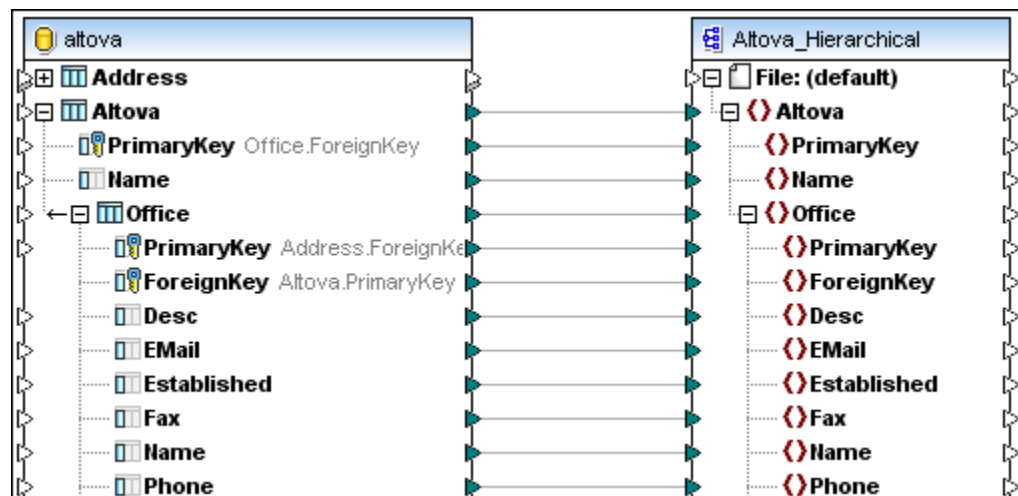
In der Beispieldatei "**DB_Altova_Hierachical.mfd**" im Ordner **...MapForceExamples** sehen Sie, wie dies beim Mappen aus einer hierarchischen Datenbank bewerkstelligt wird. Im selben Ordner finden Sie auch das Schema **Altova_Hierarchical.xsd**. Die Schemastruktur ist praktisch identisch mit der Hierarchie der Access-Datenbank. (Mit derselben Methode können Sie auch XML/SQL-Datenbanken im flachen Format, also Flat Files, mappen.)

Sie finden die MS Access-Datenbank **Altova.mdb** im Ordner **...MapForceExamples\Tutorial**.

Schemavoraussetzungen:

- Alle Tabellen, die sich auf Altova beziehen, werden als Child-Elemente des Ziel-Root-Elements angezeigt.
- Um die Tabellenbeziehungen beizubehalten, wurden alle Mappings unter der Altova-Tabelle in der Datenbankkomponente erzeugt.

Im unten gezeigten Diagramm sehen Sie das Mapping der hierarchischen Access-Datenbank auf **Altova_Hierarchical.xsd**.



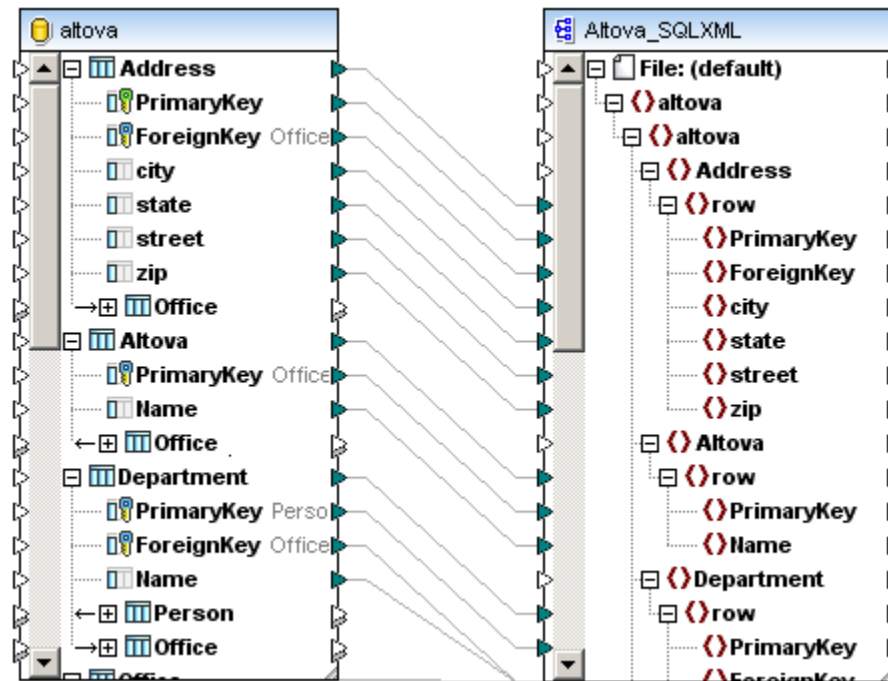
Mappen mehrerer Flat File-Tabellen auf eine XML-Ausgabedatei

Im folgenden Diagramm sehen Sie dieselbe Art von Mapping auf ein Flat File SQL/XML-Datenbankschema.

Schemavoraussetzungen:

- Die **Schemastruktur** muss den SQL/XML-Vorgaben entsprechen.
- XMLSpy ist in der Lage, mit Hilfe der Menüoption **Konvertieren | Datenbank-Schema erzeugen** eine solche SQL/XML-konforme Datei anhand einer SQL-Datenbank zu erstellen. Sie können diese dann als **Zielschema** in MapForce verwenden.
- In diesem Fall wird jeder Tabellename auf das row-Child-Element desselben Elementnamens im Schema gemappt, d.h. Address wird auf das **row**-Child-Element

des Elements **Address** gemappt.



- Beachten Sie bitte, dass im obigen Beispiel **DB_Altova_SQLXML.mfd** die Tabellenbeziehungen verloren gehen, da die Mappings anhand von mehreren verschiedenen "Root"-Tabellen erzeugt werden.

13.8 Mappings und das Root-Element von Zieldokumenten

Root-Element von XML-Zieldateien

Stellen Sie beim Erzeugen eines Mappings auf das Root-Element des Zielschemas / der XML-Zieldatei bitte sicher, dass nur ein Element bzw. ein Datensatz an die XML-Zieldatei übergeben wird, da ein XML-Dokument nur ein Root-Element haben darf.

Verwenden Sie die Filter-Komponente, um nur ein einziges Element bzw. nur einen Datensatz zu mappen.

Keine Einschränkung des Root-Elements:

Wenn Sie keine Einschränkung für das Root-Element des Zielschemas definieren, werden alle Elemente/Datensätze der Quelle zwischen dem ersten Root-Element eingefügt. Dabei werden zwar wohlgeformte, nicht aber gültige XML-Dateien erstellt.

13.9 Boolescher Vergleich von Input Nodes

Behandlung von Datentypen in Booleschen Funktionen (erstmal in MapForce 2006 SP2 eingeführt)

Während der Auswertung der Core-Funktionen: kleiner als, größer als, ist gleich, ist nicht gleich, kleiner gleich und größer gleich, hängt das ausgewertete Ergebnis zweier Input-Nodes sowohl von den Eingabewerten als auch von den für den Vergleich verwendeten Datentypen ab.

Beispiel:

Der Vergleich 'kleiner als' der Integerwerte 4 und 12 ergibt den Booleschen Wert "true", da 4 kleiner als 12 ist. Wenn die beiden Eingabestrings die Werte '4' und '12' enthalten, ist das Ergebnis der lexikalischen Analyse der Output-Wert "false", da '4' alphabetisch größer als das erste Zeichen '1' des zweiten Operanden (12) ist.

Wenn es sich bei allen "Input"-Datentypen um denselben Typ handelt, wenn z.B. alle Input Node numerische Typen oder Strings sind, wird der Vergleich für den gemeinsamen Typ durchgeführt.

Unterschiedliche Input Node-Typen (nur Version SP2):

Wenn es sich bei den Input Nodes um unterschiedliche Typen handelt, z.B. um Integer-Werte und Strings oder Strings und Datumswerte, wird die folgende Regel angewandt:

Der für den Vergleich verwendete Datentyp ist **immer der allgemeinste, d.h. der Input-Datentyp von den zwei Input-Typen mit den wenigsten Einschränkungen.**

Bevor die beiden Werte verglichen werden, werden alle Werte in einen Common Datatype konvertiert. Um beim vorhergehenden Beispiel zu bleiben: Der Datentyp "string" ist weniger restriktiv als der Wert "integer". Wenn Sie den Integer-Wert 4 mit dem String '12' vergleichen, wird der Integer-Wert 4 in den String '4' konvertiert, der anschließend mit dem String '12' verglichen wird.

Bei der Behandlung von Typen beim Vergleich von gemischten Typen wird nach den XSLT2-Richtlinien vorgegangen. Content-sensitive Typkonvertierungsstrategien werden verhindert. Der Vorteil ist, dass die Logik durch das Mapping fixiert wird und sich nicht dynamisch ändert.

Zusätzliche Überprüfungen:

Die Version 2006SP2 überprüft Mappings außerdem auf nicht kompatible Kombinationen und gibt gegebenenfalls Validierungsfehler und Warnungen aus. Beispiele sind der Vergleich von Daten mit Booleschen Werten oder "datetimes" mit numerischen Werten.

Zur Unterstützung expliziter Datentypkonvertierungen gibt es in der Core-Bibliothek die folgenden **Typkonvertierungsfunktionen**: "boolean", "number" und "string". Im oben genannten Zusammenhang eignen sich diese drei Funktionen zum Festlegen der Interpretation von Vergleichen.

core	
conversion functions	
boolean	result = boolean (arg)
number	result = number (arg)
string	result = string (arg)
logical functions	
equal	result = a equal b

Java ausgewählt

Wenn Sie diese Konvertierungsfunktionen zu Input Nodes damit in Zusammenhang stehender

Funktionen hinzufügen, können Sie dadurch den common datatype ändern und dadurch das gewünschte Ergebnis erzielen, z.B. können Sie, wenn String Nodes nur numerische Werte enthalten, einen numerischen Vergleich erzielen, indem Sie die Konvertierungsfunktion "number" (im Abschnitt "conversion" der **core**-Bibliothek) zu den einzelnen Input Nodes hinzufügen.

13.10 Prioritätskontext Node/Datenelement

Wenn Sie eine Funktion auf unterschiedliche Datenelemente in einem Schema oder einer Datenbank anwenden, muss MapForce den Context Node kennen. Alle anderen Datenelemente werden dann relativ dazu verarbeitet. Dazu wird das gewünschte Datenelement (oder der Node) als Prioritätskontext definiert.

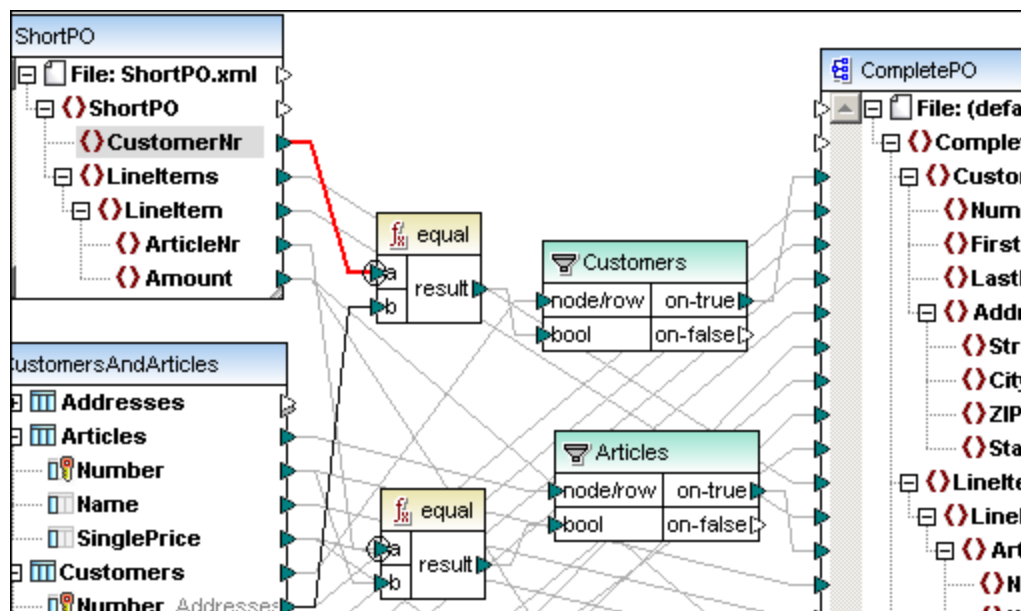
Unten sehen Sie eine vereinfachte Version der Datei **DB_CompletePO.mfd** aus dem Ordner **...MapForceExamples**.

Beachten Sie, dass es in diesem Beispiel mehrere Quellkomponenten gibt. **ShortPO** ist ein Schema, das mit einer XML-Instanzdatei verknüpft ist, während es sich bei **CustomersAndArticles** um ein Datenbankschema handelt. Die Daten aus diesen beiden Komponenten werden nun auf das Schema / die XML-Datei **CompletePO** gemappt. Das Prioritätskontext-Symbol wird durch einen Kreis gekennzeichnet.

- Das Element **CustomerNr** in ShortPO wird mit dem Datenelement **Number** in der Datenbank verglichen.
- **CustomerNr** wurde als **Prioritätskontext** definiert und in den **a**-Parameter der **equal**-Funktion gesetzt.
- Anschließend wird die Datenbank **CustomersAndArticles (einmal)** nach derselben Nummer durchsucht. Der **b**-Parameter enthält das Datenelement **Number** aus der Datenbank.
- Wird die Nummer gefunden, wird das Ergebnis an den **Booleschen** Parameter der **Filter**-Funktion (Customers) übergeben.
- Der **node/row**-Parameter übergibt die **Customer**-Daten an "on-true", wenn der Boolesche Parameter "true" ist, d.h. wenn dieselbe Nummer gefunden wurde.
- Anschließend wird der Rest der Kundendaten übergeben: Die Datenelemente **Number**, **FirstName**, **LastName** werden alle mit den entsprechenden Datenelementen im Zielschema verbunden.

Wenn Sie den **b**-Parameter der **equal**-Funktion, d.h. die Datenelement-Nr. (**Number**), als **Prioritätskontext** definieren würden, hätte dies zur Folge dass:

- MapForce die erste Zahl in den **b**-Parameter lädt
- diese mit der **CustomerNr** in **a** vergleicht, wenn sie nicht gleich ist
- die nächste Nummer in **b** lädt, diese mit **a** vergleicht und
- durch alle **Number**-Elemente in der Datenbank iteriert, um die entsprechende Nummer in ShortPO zu finden.



Prioritätskontext und benutzerdefinierte Funktionen:

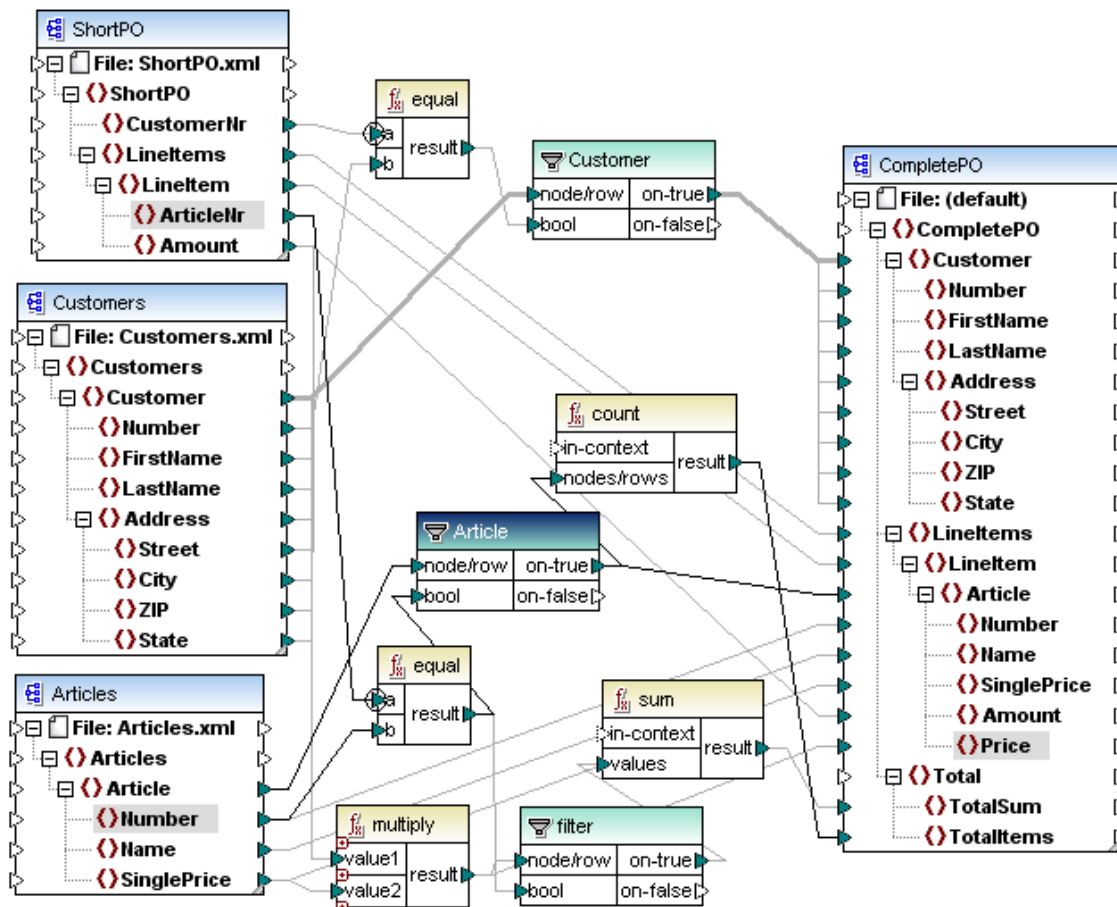
Wenn eine benutzerdefinierte Funktion vom Typ "inline" definiert wurde, (Standardeinstellung), so kann kein Prioritätskontext auf einem der Parameter der benutzerdefinierten Funktion definiert werden. Die benutzerdefinierte Funktion kann natürlich andere reguläre benutzerdefinierte Funktionen enthalten, die Parameter haben, auf denen Prioritätskontexte definiert wurden.

13.11 Zusammenführen mehrerer Dateien in einer Zielfeile

Sie können in MapForce mehrere Dateien in einer einzigen Zielfeile zusammenführen.

In diesem Beispiel werden mehrere Quellkomponenten mit unterschiedlichen Schemas zu einem Zielschema zusammengeführt. Informationen, wie Sie beliebig viele Dateien anhand desselben Schemas zusammenführen, finden Sie unter "[Dynamische Dateinamen - Input / Output](#)".

In der Datei **CompletePO.mfd** im Ordner **...\\MapForceExamples** sehen Sie, wie drei XML-Dateien in einer XML-Bestelldatei zusammengeführt werden.



Beachten Sie, dass mehrere Quellkomponentendateien in einer XML-Zielfeile - CompletePO - kombiniert werden.

- **ShortPO** ist ein Schema mit einer dazugehörigen XML-Instanzdatei und enthält nur Kundennummern- und Artikeldaten, d.h. die Einträge LineItem, Number und Amount.
- **Customers** ist ein Schema mit einer dazugehörigen XML-Instanzdatei und enthält die Kundennummern sowie die Kundendaten, also Namen und Adresse (Diese Datei enthält nur einen Kunden mit der Kundennummer 3).
- **Articles** ist ein Schema mit einer dazugehörigen XML-Instanzdatei und enthält die Artikeldaten, d.h. den Artikelnamen, die Artikelnummer und den Preis.
- **CompletePO** ist eine Schemadatei ohne Instanzdatei, da alle Daten von den drei XML-Instanzdateien geliefert werden. Aufgrund der hierarchischen Struktur dieser Datei

können alle XML-Daten darin zusammengeführt und ausgegeben werden.

Diese Schemadatei muss in einem XML-Editor wie z.B. XMLSpy erstellt werden. Sie wird nicht von MapForce generiert (obwohl dies möglich wäre, wenn Sie eine Instanzdatei für CompletePO.xml hätten).

Die Struktur von CompletePO ist eine Kombination der Strukturen der XML-Quelldateien.

Mit Hilfe der **filter**-Komponente (Customer) werden die Daten, in denen die Kundennummern sowohl in der XML-Datei "ShortPO" als auch in der XML-Datei "Customers" identisch sind, gesucht/gefiltert und die verknüpften Daten werden an die Zielkomponente "CompletePO" übergeben.

- **CustomerNr** in ShortPO wird mit Hilfe der "equal"-Funktion mit **Number** in Customers verglichen.
- Da ShortPO nur einen Kunden (Nummer 3) enthält, können nur die Kunden und Artikeldaten für den Kunden Nr. 3 an die Filterkomponente übergeben werden.
- Der **node/row**-Parameter der Filterkomponente übergibt die **Customer**-Daten (Kundendaten) an "on-true", wenn der Boolesche Parameter "true" ist, d.h. wenn dieselbe Nummer gefunden wurden, in diesem Fall Kundennummer 3.
- Die restlichen Kunden- und Artikeldaten werden über die beiden anderen Filterkomponenten an das Zielschema übergeben.

13.12 Befehlszeilenparameter

Die allgemeine Befehlszeilensyntax:

MapForce.exe *filename* [*Itarget* [*outputdir*] *options* [*I- runtimeparameters*]

- Die eckigen Klammern [...] kennzeichnen optionale Parameter.
- Die geschwungenen Klammern {...} kennzeichnen eine Parametergruppe mit mehreren Wahlmöglichkeiten.
- Das **Pipe**-Zeichen | steht für OR, z.B. /XSLT oder /JAVA

Die MapForce.exe gibt einen Exit-Code 0 zurück, wenn die Ausführung über die Befehlszeile erfolgreich war. Jeder andere Wert gibt einen Fehler an. Sie können Batch-Dateien mit dem Befehl IF ERRORLEVEL überprüfen.

filename

Die zu ladende MFD- oder MFP-Datei. **Wenn der Pfad oder der Dateiname ein Leerzeichen enthält, setzen Sie den Pfad/Dateinamen bitte in Anführungszeichen, z.B. "c:\Program Files\...\Filename"**

target

Verwenden Sie entweder BUILTIN, um das angegebene Mapping direkt auszuführen oder verwenden Sie ein Ziel für die Codegenerierung:

/BUILTIN	generiert alle Ausgaben mit Hilfe des Built-In-Ausführungsprozessors
/GENERATE	generiert anhand der aktuellen Ordneinstellungen Projektcode für alle Mappings in der Projektdatei. Die Projektdatei *.MFP muss als <i>filename</i> verwendet werden.
/XSLT	generiert XSLT1-Code
/XSLT2	generiert XSLT2-Code
/XQuery	generiert XQuery 1.0-Code
/JAVA	generiert eine JAVA-Applikation
/ICS	generiert eine C#-Applikation unter Verwendung der Konfiguration der Mapping-Einstellungen
/ICS: <i>csoptions</i>	generiert die C#-Applikation unter Verwendung der definierten Optionen

csoptions { **VS2010** | **VS2008** | **VS2005** }

VS2010 generiert eine C#-Applikation mit der Solution-Datei für Visual Studio 2010

VS2008 generiert eine C#-Applikation mit der Solution-Datei für Visual Studio 2008

VS2005 generiert eine C#-Applikation mit der Solution-Datei für Visual Studio 2005

/CPP generiert eine C++-Applikation unter Verwendung der Konfiguration der Mapping-Einstellungen

/CPP:
cppoptions generiert die C++-Applikation unter Verwendung der definierten Optionen

cppoptions { **VC10** | **VC9** | **VC8** }, { **MSXML** | **XERCES** | **XERCES3** }, { **LIB** | **DLL** }, { **MFC** | **NoMFC** }

VC10 generiert eine C++-Applikation mit der Solution-Datei für Visual Studio 2010

VC9 generiert eine C++-Applikation mit der Solution-Datei für Visual Studio 2008

VC8 generiert eine C++-Applikation mit der Solution-Datei für Visual Studio 2005

MSXML generiert C++-Code unter Verwendung von MSXML 6.0

XERCES generiert C++-Code unter Verwendung von Apache Xerces 2.x (2.6 und höher)

XERCES3 generiert C++-Code unter Verwendung von Apache Xerces 3.x

LIB generiert C++-Code für statische Bibliotheken

DLL generiert C++-Code für dynamic-linked-libraries

MFC generiert C++-Code, der MFC (Microsoft Foundation Classes) unterstützt

NoMFC generiert C++-Code ohne Unterstützung von MFC

outputdir

Das Verzeichnis in der das generierte Mapping bei der Codegenerierung gespeichert werden soll. *outputdir* ist optional. Wenn kein Ausgabepfad angegeben wird, wird das aktuelle Verzeichnis/das Arbeitsverzeichnis verwendet.

Bei der Ausführung eines Mappings mit dem Parameter */BUILTIN* wird *outputdir* als das Basisverzeichnis für die relativen Input- oder Output-Dateinamen verwendet (mit "relativ" ist gemeint, dass nur der Dateiname und nicht der komplette Pfad beginnend mit dem Laufwerksnamen angegeben wird). Falls nicht anders angegeben, sind die relativen Dateinamen relativ zum Arbeitsverzeichnis bzw. zum aktuellen Verzeichnis.

options

Dient zum Definieren verschiedener Optionen:

/STYLEVISION:outputs Generiert mittels StyleVision formatierte Ausgabedateien. Muss zusammen mit dem BUILTIN-Parameter verwendet werden und hat keine Auswirkung auf die Codegenerierung. Es muss mindestens eines der Ausgabeformate, z.B. HTML angegeben werden.

outputs HTML,RTF,PDF,DOCX

/	verwendet die globalen Ressourcen, die in
GLOBALRESOURCEFILE	der angegebenen Datei für globale
<i>grfilename</i>	Ressourcen definiert sind
/	verwendet die globale
GLOBALRESOURCECON	Ressourcenkonfiguration
FIG <i>grconfig</i>	
/LIBRARY <i>libname</i>	Wird zusammen mit /BUILTIN oder einer Codegenerierungszielsprache verwendet, um zusätzliche Funktionsbibliotheken zu definieren. Diese Option kann mehrmals verwendet werden, um mehrere Bibliotheken zu laden. Diese Bibliotheken werden vorübergehend (nur für diese eine Ausführung) aus Extras ->Optionen->Bibliotheken zu den Bibliotheken hinzugefügt.
/LOG <i>logfilename</i>	Generiert eine log-Datei. <i>logfilename</i> kann ein vollständiger Pfad sein, z.B. das Verzeichnis und der Dateiname der Log-Datei, doch muss das Verzeichnis bei Angabe eines vollständigen Pfads vorhanden sein, damit die Log-Datei generiert werden kann. Wenn nur der Dateiname angegeben wird, so wird die Datei im Verzeichnis <i>outputdir</i> gespeichert.

runtimeparameters

Definiert Werte für Input-Parameterkomponenten und Dateinamen, die nach Komponenten benannt sind. Kann nur bei Verwendung des **/BUILTIN** Parameters verwendet werden und hat keine Auswirkung bei der Codegenerierung. Alle darauf folgenden Parameter wirken sich nur auf "Input-Parameter"/Komponenten aus. Beachten Sie, dass die Runtime-Parameter von den vorhergehenden Optionen immer durch **/-** getrennt sind.

/name value Setzt den Input-Parameter mit der Bezeichnung *name* auf den Wert *value*. Dies kann für jede einzelne Komponente, für die Werte definiert werden sollen, wiederholt werden.

Wenn *name* der Name einer Input-Parameterkomponente im Hauptmapping ist, definiert *value* den Wert für diese Komponente zur Laufzeit.

Wenn *name* der eindeutige Name einer Quell- oder Zielstrukturkomponente im Hauptmapping ist, so definiert *value* den für diese Komponente zu verwendenden Dateinamen (oder die URL oder die globale Ressource). Wildcard-Dateinamen (*, ?) können nur dann für Quellkomponenten verwendet werden, wenn bereits in den Komponenteneigenschaften im Feld "Input-Datei" ein Wildcard-Zeichen für diese Komponente definiert wurde.

Quell- oder Zielkomponenten, die keinen eindeutigen Namen haben, können über die Befehlszeile nicht referenziert werden. Wenn eine Quell- oder Zielkomponente denselben Namen wie eine Input-Parameterkomponente hat, kann in der Befehlszeile nur ein Wert für die Input-Parameterkomponente definiert werden.

Anmerkungen:

Wenn Sie einen relativen Pfad in einen der Befehlszeilenparameter eingeben, so wird dies als relativ zum Arbeitsverzeichnis betrachtet, d.h. relativ zum aktuellen Verzeichnis der Applikation, die MapForce aufruft. Dies gilt für den Pfad der MFD-Datei *filename*, *outputdir*, *logfilename* und *globalresourcefilename*.

Wenn Sie einen absoluten Pfad definieren, wird das Arbeitsverzeichnis ignoriert. Der absolute Pfad wird so wie angegeben verwendet.

Vermeiden Sie in der Befehlszeile die Verwendung des End Backslash und schließenden Anführungszeichens: `\` z.B. `"C:\My directory\"`. Diese beiden Zeichen werden vom Befehlszeilenparser als Literalzeichen (doppeltes Anführungszeichen) interpretiert. Verwenden Sie den doppelten Backslash `\\`, wenn Leerzeichen in der Befehlszeile vorkommen und Sie die Anführungszeichen benötigen (`"c:\My Directory\\"`) oder verwenden Sie nach Möglichkeit gar keine Leerzeichen und Anführungszeichen z.B. `c:\MyDirectory`.

Beispiele:

MapForce.exe *filename* startet MapForce und öffnet die durch *filename* definierte Datei.

I) Alle XSLT-Dateien und Log-Datei ausgeben.

MapForce.exe *filename.mfd* **/XSLT** *outputdir* **/LOG** *logfilename*

II) Java-Applikation generieren und Log-Datei ausgeben.

MapForce.exe *filename.mfd* **/JAVA** *outputdir* **/LOG** *logfilename*

III) C#-Applikation generieren und Log-Datei ausgeben.

MapForce.exe *filename.mfd* **/ICS** *outputdir* **/LOG** *logfilename*

IV) C++-Applikation anhand der in den Applikationsoptionen definierten Codegenerierungseinstellungen generieren und Log-Datei ausgeben.

MapForce.exe *filename.mfd* **/ICPP** *outputdir* **/LOG** *logfilename*

V) C++-Applikation unter Verwendung der /ICPP-Option generieren, wobei die C++ Compiler-Optionen außer Kraft gesetzt werden.

MapForce.exe *filename.mfd* **/ICPP:(MSXML|XERCES),(LIB|DLL),(MFC|NoMFC)** *outputdir* [**/LOG** *logfilename*]

MapForce.exe *filename.mfd* **/ICPP:MSXML,LIB,MFC**

Generiert die C++-Applikation unter Verwendung jeweils der ersten Wahl - in diesem Beispiel:

- für C++ kompilieren
- MSXML verwenden
- Code für statische Bibliotheken generieren
- der generierte Code muss MFC unterstützen

MapForce.exe *filename.mfd* **/ICPP:XERCES,DLL,NoMFC** *outputdir* **/LOG** *logfilename*

Generiert die C++ Applikation unter Verwendung jeweils der zweiten Wahl - in diesem Beispiel:

- für C++ kompilieren
- XERCES verwenden
- Code für DLL-Bibliotheken generieren
- Code generieren, der MFC nicht unterstützt
- im outputdir eine Log-Datei mit dem Namen logfilename erstellen

VI)

Alle Ausgabedateien (XML-Zieldokument und Datenbanken) mit Hilfe des Built-In-Ausführungsprozessors generieren.

MapForce.exe *filename.mfd* **/BUILTIN** *outputdir*

VII)

Alle Ausgabedateien mit dem Built-in-Ausführungsprozessor generieren und die Input- und Output-**Dateinamen** bei Ausführung des Befehls generieren. Zusätzlich dazu eine Log-Datei generieren.

Mapforce.exe *filename.mfd* **/BUILTIN /LOG** *logfilename* **/- /InputFileName** *inputfilename* **/OutputFileName** *Outputfilename*.

Anmerkung: Die **Parameter** /InputFileName und /OutputFileName sind die Namen der speziellen Input-Komponenten im MapForce-Mapping, die als Parameter bei der Ausführung in der Befehlszeile dienen.

Bei Verwendung des Built-in-Ausführungsprozessors ersetzen die Werte aus der Befehlszeile die im Mapping definierten Werte. Informationen zum Definieren der Input- und Output-Parameter bzw. -Datei finden Sie unter [Befehlszeile - Definieren von Input/Output-Dateien](#).

VIII) Alle Ausgabedateien mit Hilfe des Built-In-Ausführungsprozessors generieren und alle globalen Ressourcen der globalen Ressourcendatei für die definierte Konfiguration verwenden.

Mapforce.exe *filename.mfd* **/BUILTIN** *outputdir* **/IGLOBALRESOURCEFILE** *globalresourcefilename* **/IGLOBALRESOURCECONFIG** *configurationname*

IX) Anhand der aktuellen Projekteinstellungen Projektcode für alle Mappings in der Projektdatei generieren

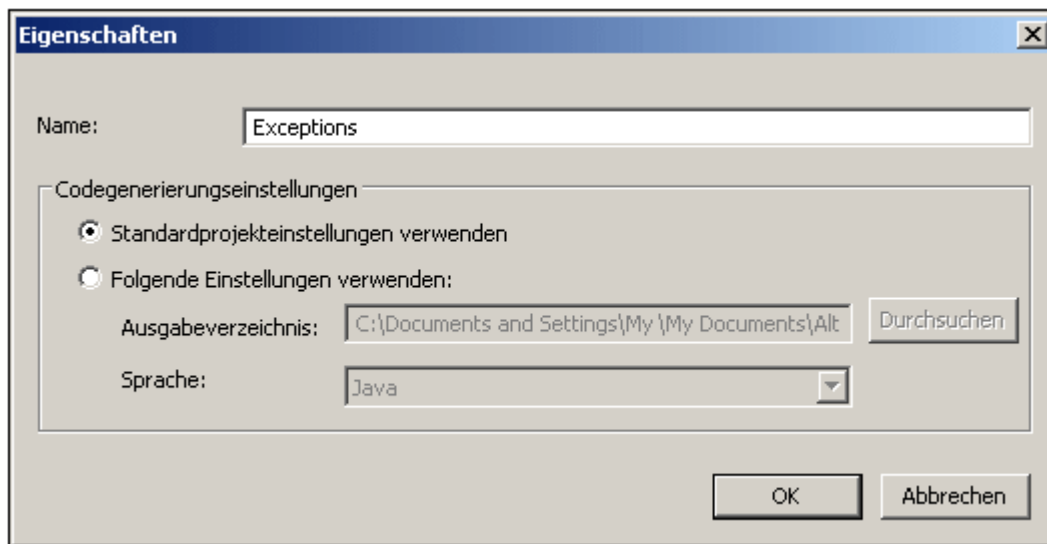
MapForce.exe *filename.mfp* /GENERATE /LOG *logfilename*

Bei der Generierung von Code für das gesamte Projekt muss der Projektname, z.B.

MapForceExamples.mfp, als *filename* verwendet werden.

Die Codegenerierungssprache sowie der Ausgabepfad für die Mappings in den einzelnen Ordnern stammen aus den Ordneigenschaften (Dialogfeld "Eigenschaften") für jeden einzelnen Ordner, wenn **kein Ausgabeverzeichnis** (*outputdir*) definiert ist.

Wenn beim Generieren von Projektcode unter Verwendung von GENERATE der Parameter *outputdir* verwendet wird, so wird der Pfad des Verzeichnisses *Outputdir* des **Projekt-Rootverzeichnisses** durch den Pfad überschrieben, den Sie in der Befehlszeile angeben. Dies ist das Verzeichnis, das verwendet wird, wenn Sie beim Hinzufügen eines neuen Ordners zum Projekt im Dialogfeld "Eigenschaften" dieses Ordners das Optionsfeld "Standardprojekteinstellungen verwenden" aktiviert haben.



X) Projektcode für alle Mappings in der Projektdatei in Java generieren

MapForce.exe *filename.mfp* /JJAVA /LOG *logfilename*

Die in den Ordneigenschaften definierten Einstellungen für die Codegenerierungssprache werden ignoriert und Java wird für alle Mappings verwendet.

XI)

Mit der ausführbaren JAR-Datei Ausgabedateien eines zuvor kompilierten (Java)-Mapping-Projekts generieren; und die Input- und Output-Datei in der Befehlszeile definieren.

java -jar *mappingfile.jar* /InputFileName *inputfilename* /OutputFileName *outputfilename*

Anmerkung: Die **Parameter** /InputFileName und /OutputFileName sind die Namen der speziellen Input-Komponenten im MapForce-Mapping, die als Parameter bei der Ausführung in der Befehlszeile dienen.

Informationen zum Definieren der Input- und Output-Parameter für die Dateinamen finden Sie unter [Befehlszeile - Definieren von Input/Output-Dateien](#).

XII)

Erzeugen formatierter Berichte zu einem Mapping mit Hilfe von StyleVision und Generierung eines HTML- und PDF-Dokuments des Mappings.

MapForce.exe *filename.mfd* **/BUILTIN** *outputdir* **/STYLEVISION:SVOptions** **/LOG** *logfilename*

MapForce.exe myMapping.mfd **/BUILTIN** **/STYLEVISION:HTML,PDF** **/LOG** mylogfile.txt

XIII)

Verwendung eines vorhandenen Mapping-Projekts (FNAP.mfd), wobei jedoch eine andere Input-Instanzdatei verwendet und eine andere Output-Datei generiert wird, als ursprünglich im Mapping-Projekt definiert wurde.

In diesem Beispiel machen wir uns die Tatsache zunutze, dass Komponente eigene Namen haben, anhand derer sie eindeutig identifiziert werden können. Mit Hilfe dieser Namen können ihnen bei der Ausführung über die Befehlszeile unterschiedliche Instanz-/Output-Dateien zugewiesen werden..

Mapforce.exe *filename.mfd* **/BUILTIN** **/LOG** *logfilename* **I-** *runtimeparameters*
/Parameter-name1 *parameter-value1* **Parameter-name2** *parameter-value2*.

Mapforce.exe FNAP.mfd **/BUILTIN** **/LOG** *log.txt* **I-** *Altova_Hierarchical*
Altova_hierarchical2.xml *Altova_Hierarchical_targ* *Altova_out.xml*

Anmerkung:

Altova_hierarchical2.xml wird als die neue Input-Instanz der Quellkomponente

Altova_Hierarchical verwendet

Altova_out.xml ist der Name der neuen Output/Zieldatei der Zielkomponente

Altova_Hierarchical_targ

Eine Beschreibung dazu finden Sie unter: [Befehlszeile - Komponentennamen](#).

13.13 Befehlszeile - Definieren von Input-Parametern

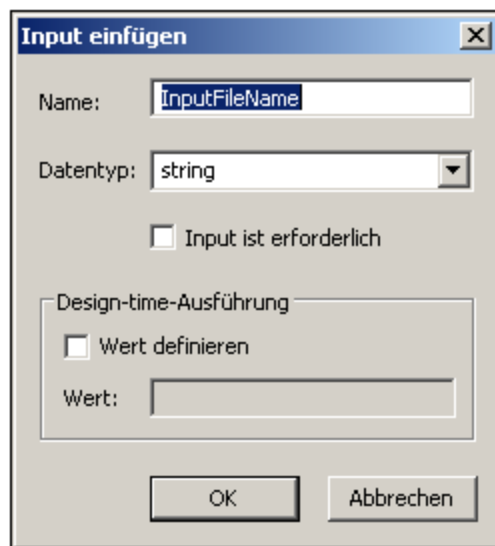
MapForce gestattet das Erstellen von "Input"-Komponenten, die als **Parameter** in der Befehlszeilenausführung des kompilierten Mappings fungieren können. Mit Hilfe dieser Input-Komponenten können Sie beim Ausführen des Mappings über die Befehlszeile definieren, welchen Inhalt der Parameter bei der Ausführung der Befehlszeile haben soll und welche Werte in der Vorschau der Ausgabe in MapForce angezeigt werden sollen. Der Inhalt des Parameters kann ein Dateiname oder jeder beliebige Wert, der an das Mapping übergeben werden muss, sein.

Diese spezifische Art von "Input"-Komponente kann nicht **innerhalb** einer benutzerdefinierten Funktion verwendet werden, sondern steht nur im Hauptmapping zur Verfügung.

Neben den Input-Komponenten können in der Befehlszeile auch alle Quell- und Zielkomponenten, die einen eindeutigen Namen haben, als Parameter verwendet werden. Siehe dazu [Befehlszeile](#). Mit Hilfe von Input-Komponenten können Dateinamen an das Mapping übergeben werden, indem Sie diese mit dem "Datei"-Input Node von Komponenten verbinden (wie im Beispiel FileNamesAsParameters.mfd gezeigt). In vielen Fällen ist es jedoch einfacher, die Komponentennamen direkt zu verwenden.

So definieren Sie eine Input-Komponente / einen Befehlszeilenparameter:

1. Verwenden Sie die Menüoption **Funktion | Input-Komponente einfügen**, um die Komponente einzufügen.
Daraufhin wird das Dialogfeld "Input einfügen" geöffnet.
2. Geben Sie einen Namen für die Funktion ein, z.B. **InputFileName** und wählen Sie den gewünschten Datentyp aus, z.B. string.



3. Klicken Sie auf OK, um die Definition fertigzustellen.

Die anderen Felder in diesem Dialogfeld dienen zum Definieren der Werte, die bei Ausführung des Mappings auf dem Register "Ausgabe" verwendet werden sollen. Sie werden bei der Ausführung über die Befehlszeile nicht benötigt. Nähere Informationen dazu finden Sie unter [Input-Parameter - Standardeinstellungen und Einstellungen für die Vorschau](#).

Definieren der Werte für die Ausführung über die Befehlszeile:

Bei Verwendung unseres Beispiels könnte die Befehlszeile in etwa wie unten gezeigt aussehen:
Mapforce.exe FindHighestTemperatures.mfd **/BUILTIN /LOG** log.txt **/- /count** 5

In diesem Beispiel:

- wird die Mapping-Datei FindHighestTemperatures.mfd
- wird mit Hilfe des Built-in-Ausführungsprozessors die Mapping-Ausgabe generiert
- wird eine Datei log.txt für die Ausführungsmeldungen erstellt
- wird der Wert 5 für die Input-Komponente mit den Namen **count** angegeben

Der **/-** Switch (Runtime-Parameter) kann nur bei Verwendung des BUILTIN-Ausführungsprozessors verwendet werden und hat bei der Generierung von Code keine Auswirkung. Er gibt an, dass sich alle darauf folgenden Parameter nur auf die speziellen "Input"-Parameter/Komponenten auswirken.

Aufrufen kompilierter Programme über die Befehlszeile:

Um Input- / Output-Dateinamen für das kompilierte Mapping FileNameAsParameters.mfd anzugeben, gehen Sie folgendermaßen vor:

1. Generieren Sie in MapForce Java-Code für das Mapping: **Datei | Code generieren in | Java.**
2. Machen Sie Ihren generierten Java-Code mit Hilfe der IDE Ihrer Wahl zu einer ausführbaren JAR-Datei, z.B. myMapping.jar.
3. Geben Sie die Befehlszeile ein:
java -jar myMapping.jar **/count** 5

Platzhalterzeichen in der Befehlszeilenausführung (Java):

Beim Aufrufen eines generierten und kompilierten Java-Programms über die Befehlszeile müssen die Platzhalter unbedingt in Anführungszeichen gesetzt werden.

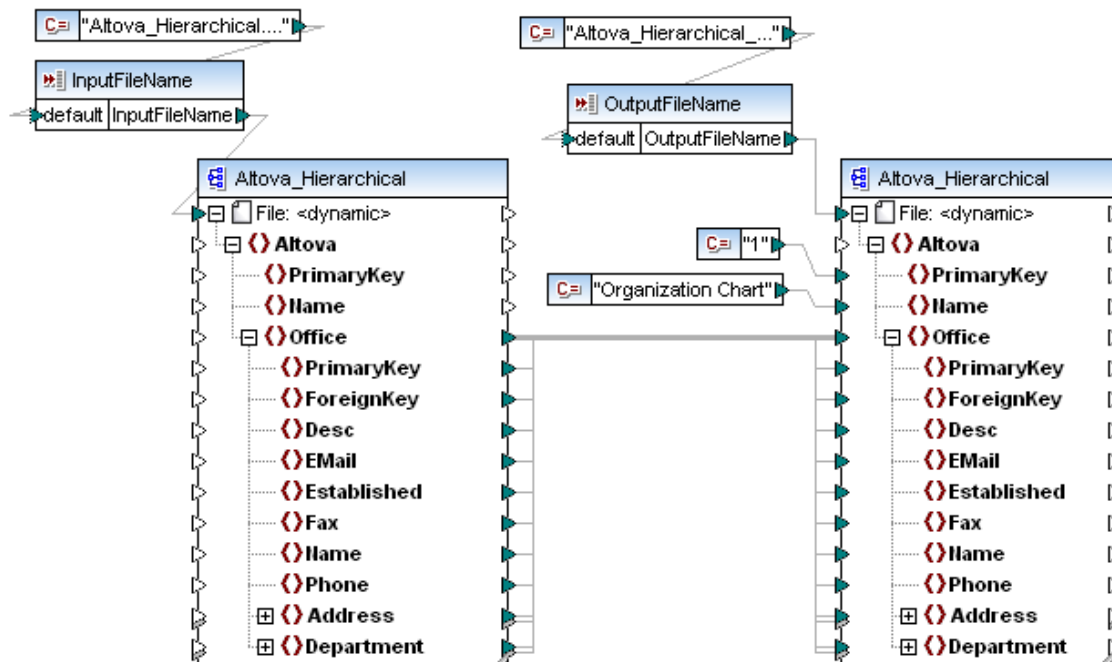
Richtige Verwendung:

```
java -jar myMapping.jar /InputFileName "altova-*.xml"
```


13.14 Input-Parameter - Standardeinstellungen und Einstellungen für die Vorschau

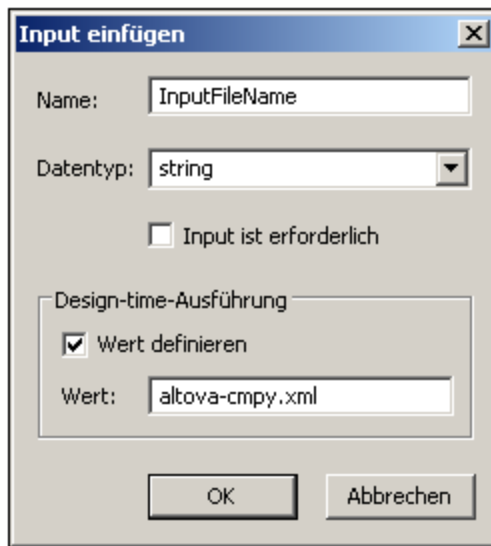
Im unten gezeigten Mapping (**FileNamesAsParameters.mfd** im Ordner ...\\MapForceExamples) werden zwei Input-Komponenten verwendet, eine liefert den Namen der Input-Datei und die andere definiert den Namen der Output-Datei für das Datenelement "Datei: <dynamisch>" der einzelnen Komponenten.

Die Komponenten **InputFileName** und **OutputFileName** sind die [Input-Komponenten](#) im Mapping, mit Hilfe derer diese als Dateinamenparameter bei der Ausführung über die Befehlszeile verwendet werden können.



So definieren Sie eine Input-Komponente:

1. Verwenden Sie die Menüoption **Funktion | Input-Komponente einfügen**, um die Komponente einzufügen. Daraufhin wird das Dialogfeld "Input einfügen" geöffnet.
2. Geben Sie einen Namen für die Funktion ein, z.B. **InputFileName** und wählen Sie den gewünschten Datentyp aus, z.B. string.



3. Aktivieren Sie das Kontrollkästchen "Wert definieren" und geben Sie den Wert/String, den Sie in der Ausgabevorschau verwenden möchten, in das Textfeld ein, z.B. altova-cmpy.xml.
4. Klicken Sie auf die Schaltfläche "OK" und anschließend auf die Schaltfläche "Ausgabe", um das Ergebnis zu sehen.
Die Datei "altova-cmpy.xml" wird als Quelldatei für das Mapping verwendet und die gemappten Daten werden im Ausgabefenster angezeigt.

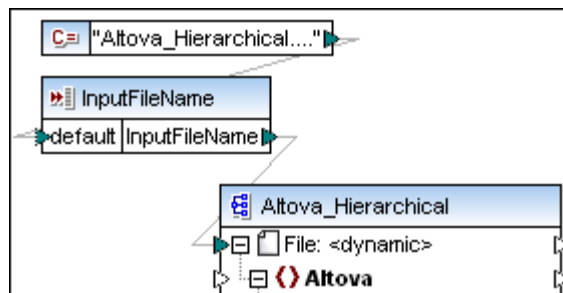
Der Wert, der im Bereich "Design-time-Ausführung" in das Feld "**Wert**" eingegeben wurde, wird **nur** bei der **Vorschau** der Ergebnisse auf dem Register **Ausgabe** angewendet. Er wird nicht für die Codegenerierung oder die Mapping-Ausführung über die Befehlszeile verwendet.

So definieren Sie einen Standardwert:

Nachdem Sie die Input-Komponente definiert und auf die Schaltfläche "OK" geklickt haben, wird die Komponente wie unten gezeigt im Mapping-Bereich angezeigt. Beachten Sie, dass auf der linken Seite der Eintrag "default" angezeigt wird, während der Name des anderen mapbaren Eintrags der Komponententennamen ist.

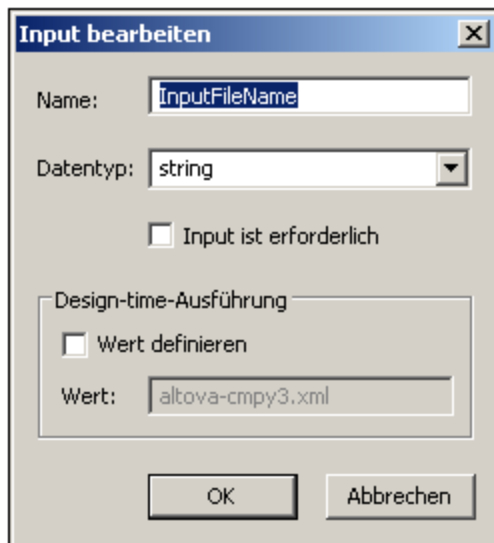


1. Sie können den Standardwert, d.h. den Wert von "default", wie oben gezeigt, mit Hilfe einer Konstantenkomponente angeben, z.B. Altova_Hierarchical.xml.
2. Erstellen Sie die Konstantenkomponente und verbinden Sie sie mit dem default-Element der Input-Komponente.



So verwenden Sie den Standardwert einer Input-Komponente:

1. Doppelklicken Sie auf die Input-Komponente und deaktivieren Sie die Kontrollkästchen "Input ist erforderlich" und "Wert definieren".



2. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen. Die Daten aus der Datei Altova_Hierarchical.xml werden nun für die Vorschau verwendet.

Generieren von XSLT 1.0- oder XSLT 2.0-Code:

Der Wert aus dem Textfeld "Wert" wird in die Batch-Datei **DoTransform.bat** geschrieben. Diese Batch-Datei wird automatisch für die Ausführung im AltovaXML-Prozessor generiert. Wenn Sie eine andere Input- (oder Output)-Datei verwenden möchten, können Sie den Namen in der Batch-Datei ändern.

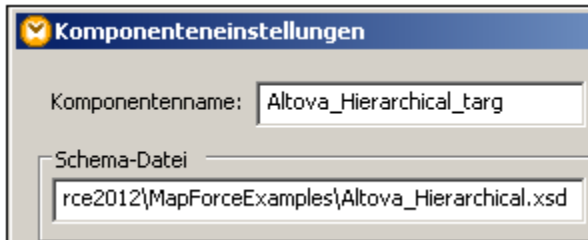
Wenn kein Wert in das Textfeld "Wert" eingegeben wurde, so wird der Standardwert, der im generierten XSLT-Code vorhanden ist, verwendet.

Verwendung von Standardparametern/Standardwerten in der Ausführung über die Befehlszeile:

Damit der Standardparameter aus der Befehlszeile verwendet wird, muss das Kontrollkästchen "Input ist erforderlich" (vor der Codegenerierung) **deaktiviert werden**. Geben Sie **mapping.exe** ein, damit der generierte Code den Standardparameter aus der Befehlszeile verwendet.

13.15 Komponentennamen

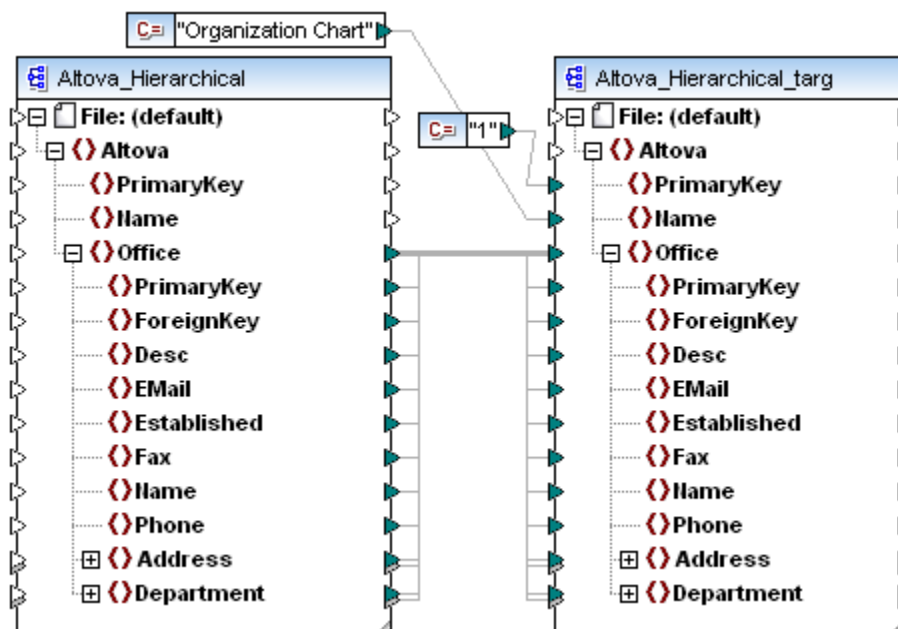
Alle dateibasierten Komponenten in MapForce können einen benutzerdefinierten Komponentennamen haben. Der Komponentennamen wird im obersten Feld des Dialogfelds "Komponenteneinstellungen" angezeigt.



Der Komponentennamen wird in den folgenden Situationen verwendet:

- Die von der Komponente verwendete Input- oder Output-Datei kann zur Zeit der Ausführung über die Befehlszeile (mit Hilfe des [/BUILTIN](#) Switch) oder über FlowForce definiert werden, ohne dass die Input-Komponente wie unter [Befehlszeile - Definieren von Input-Parametern](#) beschrieben, verwendet werden muss.
- Der Name der generierten XSLT/XQuery Scripts wird vom Komponentennamen abgeleitet.
- Wenn Sie über die Befehlszeile oder über FlowForce auf eine Komponente zugreifen möchten, so muss der Komponentennamen eindeutig sein. Nähere Informationen dazu finden Sie unter [Komponente](#).

Das unten gezeigte Mapping ist eine vereinfachte Version des Mappings **FileNamesAsParameters.mfd** aus dem Ordner ...MapForceExamples.



In dem vereinfachten Beispiel werden keine separaten Input-Komponenten zum Definieren der Instanzdateien verwendet. Diese Dateien/Parameter werden in der Befehlszeile unter Verwendung eindeutiger Komponentennamen definiert:

- Der Name der Quellkomponente ist **Altova_Hierarchical**
- Der Name der Zielkomponente ist **Altova_Hierarchical_targ**

Angabe der Input- und Output-Parameter für die Ausführung über die Befehlszeile:

Geben Sie die Befehlszeile in der unten gezeigten Form ein.

Mapforce.exe FNAP.mfd /**BUILTIN** /**LOG** log.txt **/-** /**Altova_Hierarchical**

Altova_hierarchical2.xml /**Altova_Hierarchical_targ** Altova_out.xml

In diesem Beispiel:

- wird die Mapping-Datei FNAP.mfd verwendet
- wird der Built-in-Ausführungsprozessor zum Generieren der Mapping-Ausgabe verwendet
- wird eine log.txt-Datei mit den Ausführungsmeldungen erstellt
- wird als Input-Instanz der Quellkomponente **Altova_Hierarchical** die Datei **Altova_hierarchical2.xml** verwendet
- wird **Altova_out.xml** als Instanzdatei der Zielkomponente **Altova_Hierarchical_targ**, verwendet, in die das Mapping-Ergebnis geschrieben wird

Der **/-** Switch (Laufzeitparameter) kann nur bei Verwendung des BUILTIN-Ausführungsprozessors verwendet werden. Er hat keine Auswirkung auf die Codegenerierung. Dieser Switch gibt an, dass alle darauf folgenden Parameter nur die folgenden Parameter/Komponenten betreffen.

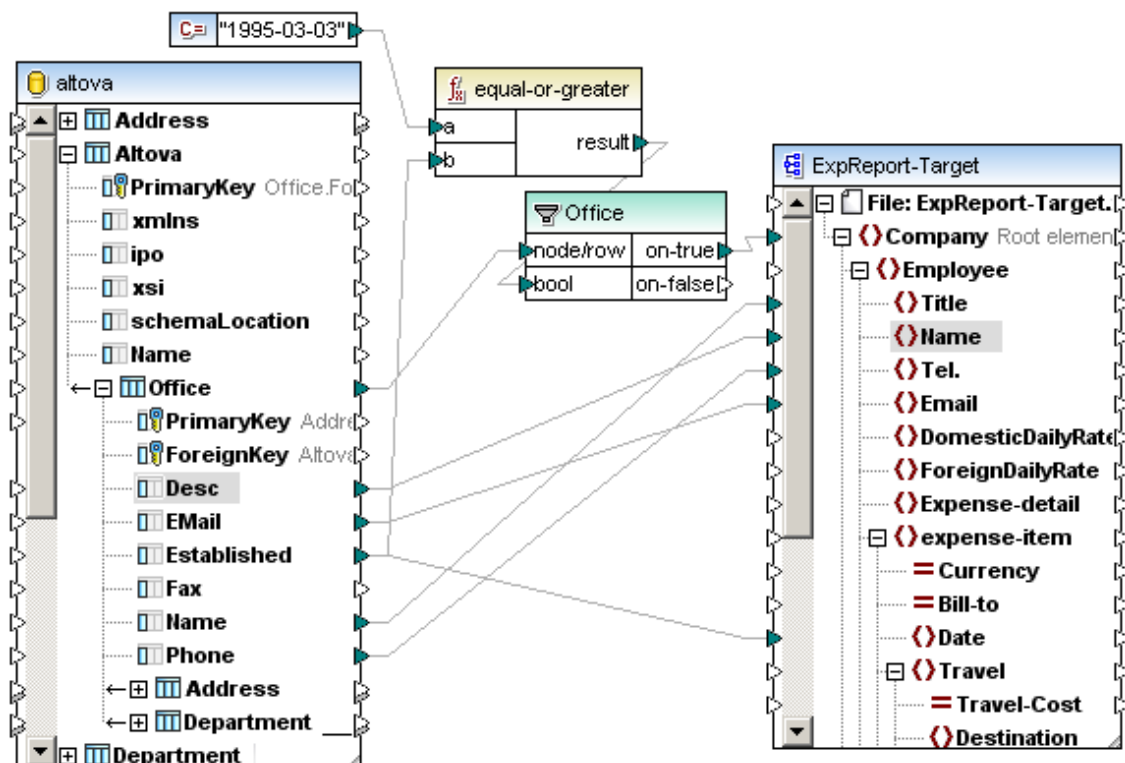
Wildcard-Dateinamen (*, ?) können nur dann für Quellkomponenten verwendet werden, wenn für diese Komponente in den Komponenteneigenschaften im Feld "Input-Datei" bereits ein Wildcard-Zeichen definiert wurde.

Quell- oder Zielkomponenten, die keinen eindeutigen Namen haben, können über die Befehlszeile nicht referenziert werden. Wenn eine Quell- oder Zielkomponente denselben Namen wie eine Input-Parameterkomponente hat, kann in der Befehlszeile nur ein Wert für den Input-Parameterkomponente definiert werden.

13.16 Filtern von Datenbankdaten nach Datum

Im unten gezeigten Beispiel sehen Sie, wie Sie mit Hilfe der Filterkomponente Datenbank-Datensätze nach einem bestimmten Datum filtern können.

- Das Feld "Established" wird als "Date/Time" Feld in der Datenbank definiert.
- Das Vergleichsdatum wird in eine Konstantenkomponente eingegeben.
- Nur wenn das Datum im Datensatz größer als 1995-03-03 ist, werden die jeweiligen "Office"-Daten von der Filterkomponente an die Zielfelder übergeben. Anmerkung: Verwenden Sie den Datentyp **"All other"** für die Konstantenkomponente.



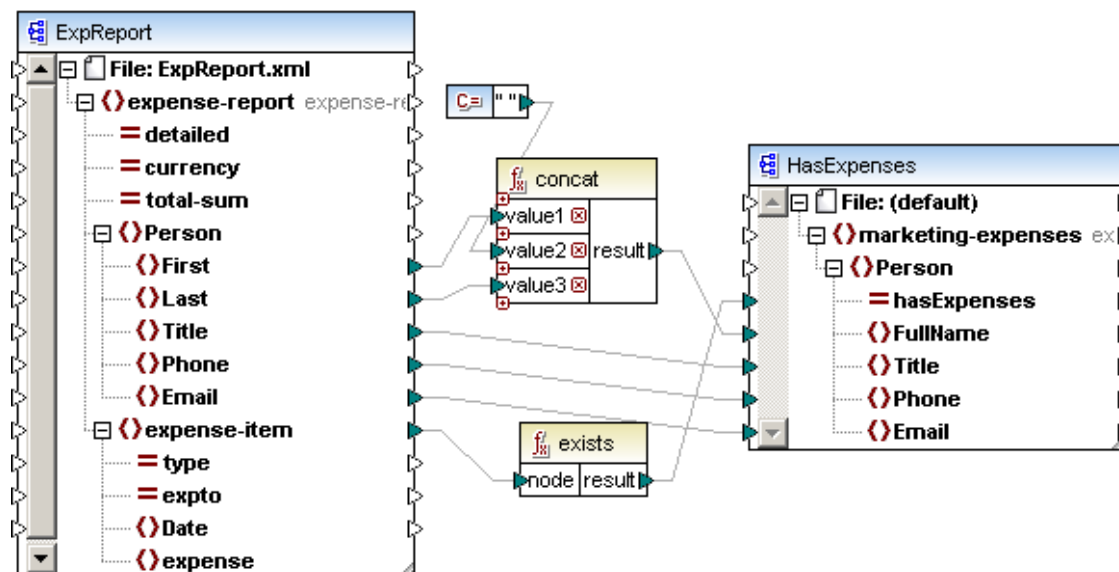
13.17 Node-Tests, Position und Gruppierung

Die Funktionen zum Testen von Nodes gestatten Ihnen, zu überprüfen, ob bestimmte Nodes in den **XML-Instanzdateien** existieren. Elemente oder Attribute, die im XML-Schema als optional definiert sind, können - müssen aber nicht - in der XML-Instanzdatei vorkommen. Mit Hilfe dieser Funktionen können Sie bestimmte Nodes überprüfen und dieses Resultat einer weiteren Verarbeitung zugrunde legen.

Exists

Gibt den Wert "true" zurück, wenn der Node vorhanden ist, andernfalls ist das Resultat "false". Die Datei "HasMarketingExpenses.mfd" im Ordner **...IMapForceExamples** enthält das unten gezeigte kurze Beispiel.

Wenn ein Datenelement "expense-item" in der XML-Quelldatei existiert, wird das Attribut "hasExpenses" in der XML/Schema-Zielfdatei auf "true" gesetzt.

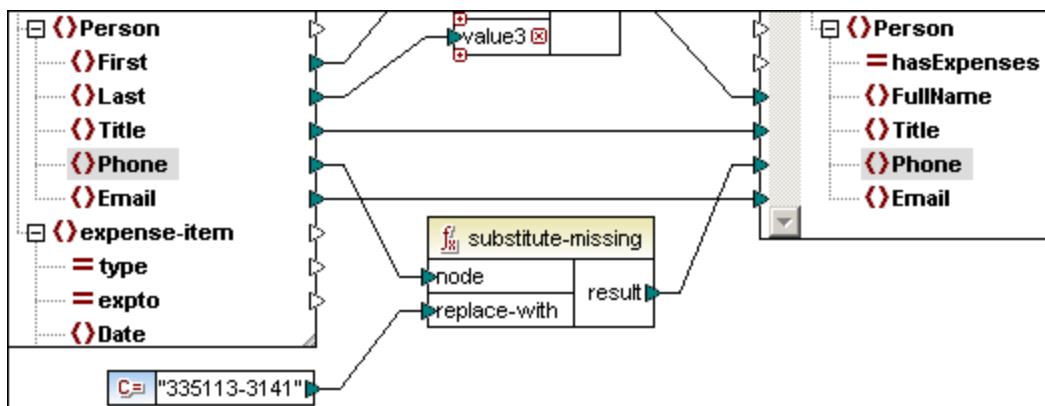


Not-exist

Gibt den Wert "false" zurück, wenn der Node existiert. Andernfalls ist das Resultat "true". Ein Beispiel zum Mappen von fehlenden Nodes finden Sie unter [Mappen fehlender Nodes - mit Hilfe von "Not-exist"](#).

substitute-missing

Diese Funktion ist eine praktische Kombination aus **exists** und einer passenden **if-else**-Bedingung. Wird verwendet, um den Inhalt des aktuellen Felds zu mappen, wenn der Node in der XML-Quelldatei existiert, andernfalls wird das Datenelement verwendet, das auf den Parameter "replace-with" gemappt ist.



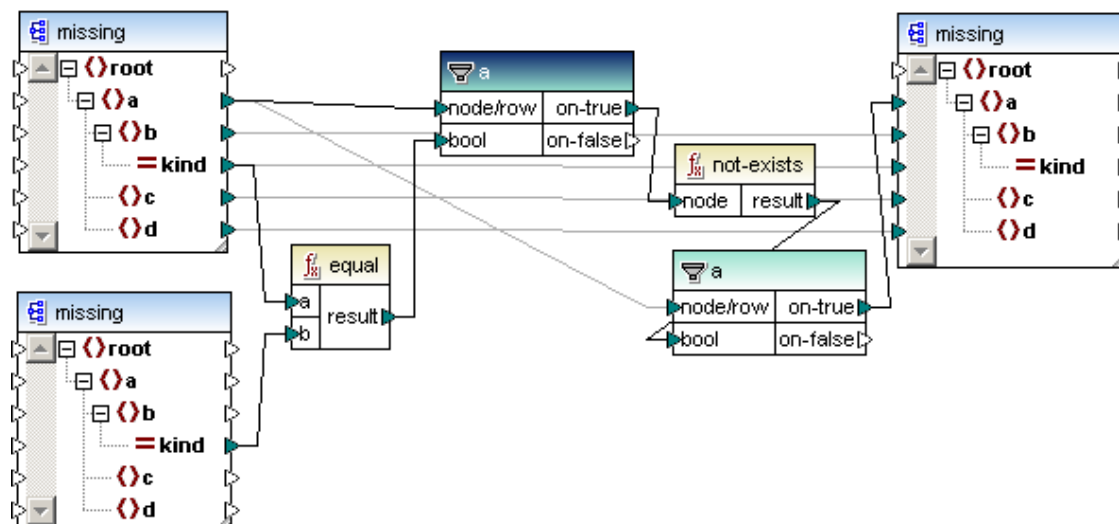
In der Abbildung oben wird die Existenz des Nodes "Phone" in der XML-Instanzdatei überprüft. Wenn dieser Node fehlt, wird der von der Konstante bereitgestellte Wert gemappt.

13.17.1 Mappen fehlender Nodes - mit Hilfe von Not-exists

Im Beispiel unten sehen Sie, wie Sie Nodes, die in einer von zwei Quelldateien fehlen, mit Hilfe der not-exists-Funktion mappen können.

Funktionsweise dieses Mappings:

- Vergleich der Nodes von zwei XML-Quelldateien
- Herausfilterung der Nodes der ersten XML-Quelldatei, die in der zweiten XML-Datei nicht vorhanden sind
- Mappen nur der fehlenden Nodes und ihres Inhalts auf die Zieldatei.



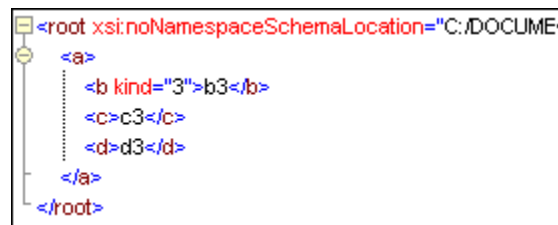
Die beiden XML-Instanzdateien werden unten angezeigt. Die Unterschiede zwischen ihnen sind die folgenden:

- **a.xml** auf der linken Seite enthält den Node `<b kind="3">`, der in **b.xml** fehlt.
- **b.xml** auf der rechten Seite enthält den Node `<b kind="4">`, der in **a.xml** fehlt.



- Die **equal**-Funktion vergleicht das Attribut **kind** der beiden XML-Dateien und übergibt das Ergebnis an den Filter.
- Die Funktion **not-exists** wird hinter dem Anfangsfilter platziert und wählt die fehlenden Nodes der beiden Quelldateien aus.
- Über den zweiten Filter werden der fehlende Node und andere Daten **ausschließlich** aus der Datei **a.xml** an die Zieldatei übergeben.

Das Ergebnis des Mappings ist, dass der in der Datei **b.xml** fehlende Node, `<b kind="3">`, an die Zielkomponente übergeben wird.

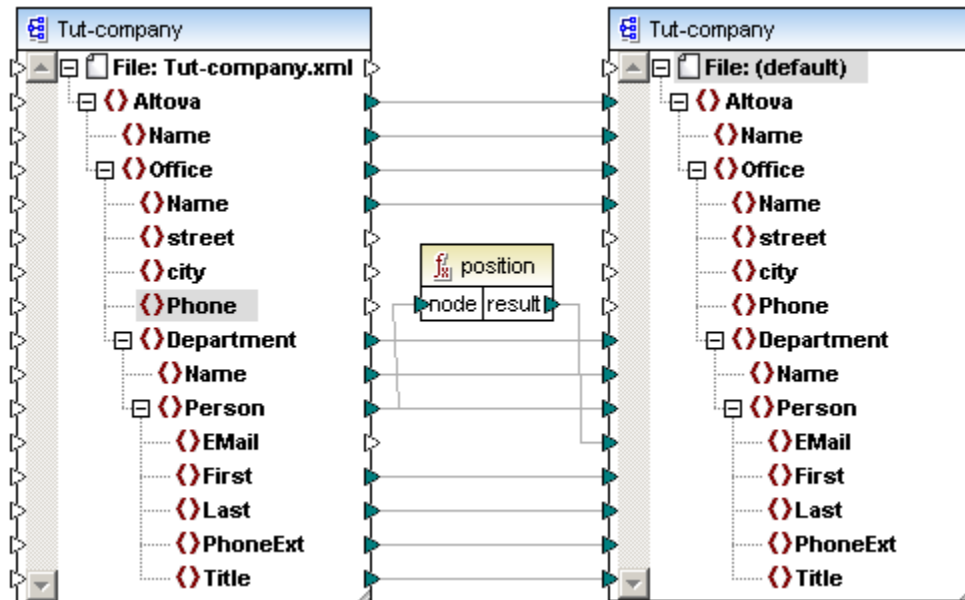


13.17.2 Position von Kontextelementen in einer Sequenz

Mit Hilfe der Positionsfunktion können Sie die Position bestimmter Nodes in einer Sequenz ermitteln oder anhand einer bestimmten Position Datenelemente herausfiltern.

Das Kontextelement wird durch das mit dem "node"-Parameter der Positionsfunktion verbundene Datenelement definiert, im Beispiel unten "Person".

Im unten gezeigten einfachen Mapping wird zu jedem Person-Element jeder Abteilung (Department) eine Positionsnummer hinzugefügt.



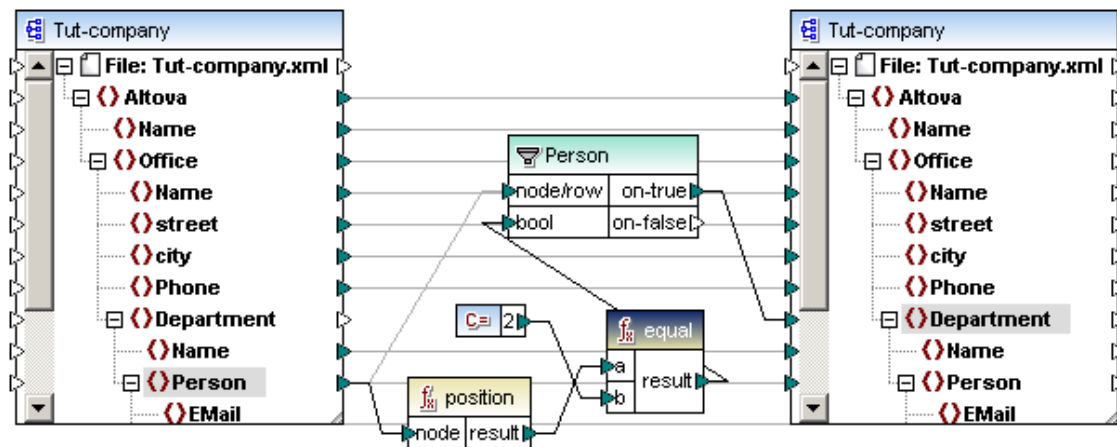
Die Positionsnummer wird für jede Abteilung im Büro zurückgesetzt.

```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>1</EMail>
      <First>Albert</First>
      <Last>Aldrich</Last>
      <PhoneExt>582</PhoneExt>
      <Title>Manager</Title>
    </Person>
    <Person>
      <EMail>2</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <PhoneExt>471</PhoneExt>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
</Office>
```

Herausfiltern bestimmter Nodes mit Hilfe der Positionsfunktion

Mit Hilfe der Positionsfunktion in Verbindung mit einem Filter können Sie nur die Nodes, die eine bestimmte Position in der Quellkomponente haben, mappen.

Es scheint, dass der Filterparameter "node/row" und die position "node"-Funktion mit demselben Datenelement der Quellkomponente verbunden werden müssen, um eine bestimmte Position dieser Sequenz herauszufiltern.



Das Ergebnis des Mappings ist Folgendes:

- Ausgabe der **zweiten** Person in jeder Abteilung (Department)
- von jedem Büro bei Altova.

```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>b.bander@microtech.com</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
  <Department>
    <Name>Sales and Marketing</Name>
    <Person>
      <EMail>e.ellas@microtech.com</EMail>
      <First>Eve</First>
      <Last>Ellas</Last>
      <Title>Art Director</Title>
    </Person>
  </Department>
  <Department>
    <Name>Manufacturing</Name>
    <Person>
      <EMail>g.gundall@microtech.com</EMail>
```

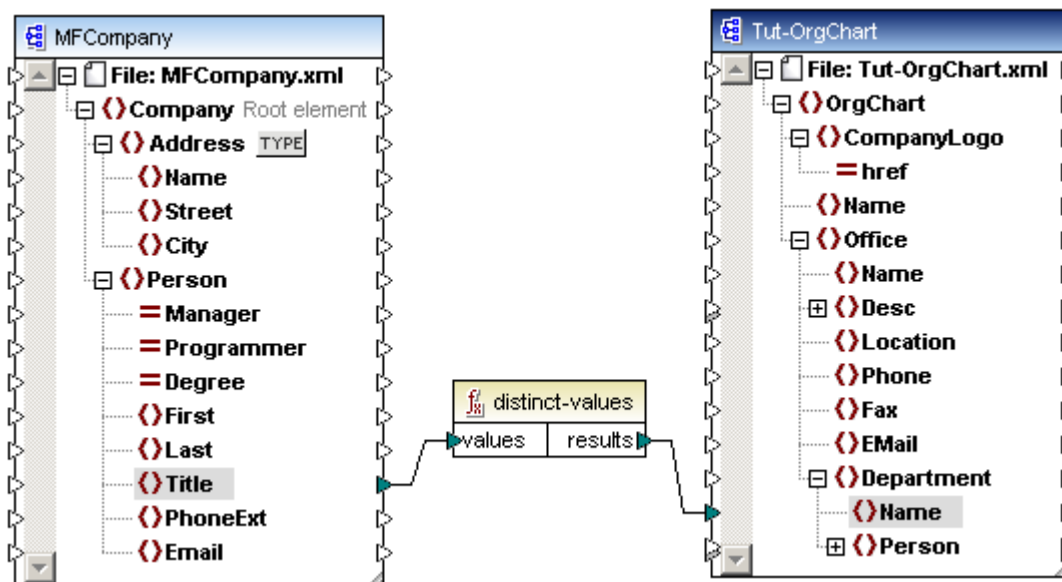
13.17.3 Gruppieren von Nodes / Node-Inhalt

MapForce unterstützt nun die Gruppierung von Nodes und ihrem Inhalt. Sie finden diese Funktionen im Fenster "Bibliotheken" im Abschnitt "Sequence functions".

distinct-values

Mit Hilfe dieser Funktion können Sie doppelt vorhandene Werte aus einem Ergebnis entfernen und die eindeutigen Datenelemente auf die Zielkomponente mappen.

Im unten gezeigten Beispiel wird der Inhalt der Quellkomponente "Title" durchsucht und jedes eindeutige Title-Element wird auf den Department / Name-Node der Zielkomponente gemappt.



Beachten Sie, dass die Reihenfolge der einzelnen Title-Elemente in der Quellkomponente beim Mappen auf die Zielkomponente beibehalten wird.

```

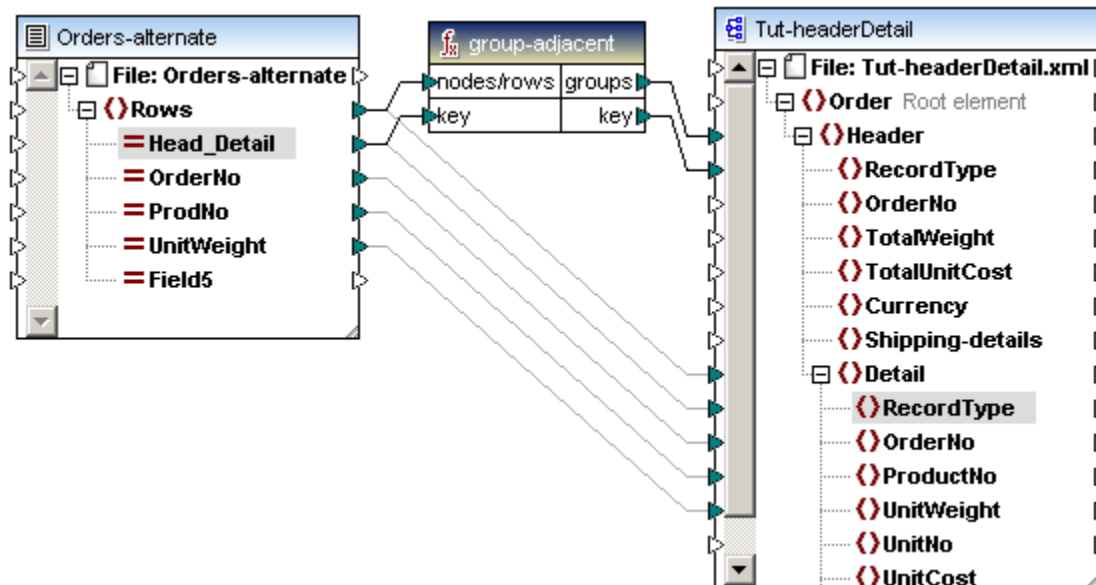
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3  <Office>
4  <Department>
5      <Name>Office Manager</Name>
6      <Name>Accounts Receivable</Name>
7      <Name>Accounting Manager</Name>
8      <Name>Marketing Manager Europe</Name>
9      <Name>Art Director</Name>
10     <Name>Program Manager</Name>
11     <Name>Software Engineer</Name>
12     <Name>Technical Writer</Name>
13     <Name>IT Manager</Name>
14     <Name>Web Developer</Name>
15     <Name>Support Engineer</Name>
16     <Name>PR & Marketing Manager US</Name>
17 </Department>
18 </Office>
19 </OrgChart>

```

group-adjacent

Gruppert die Input-Sequenz in einer Reihe von Gruppen, wobei jede Gruppe identisch

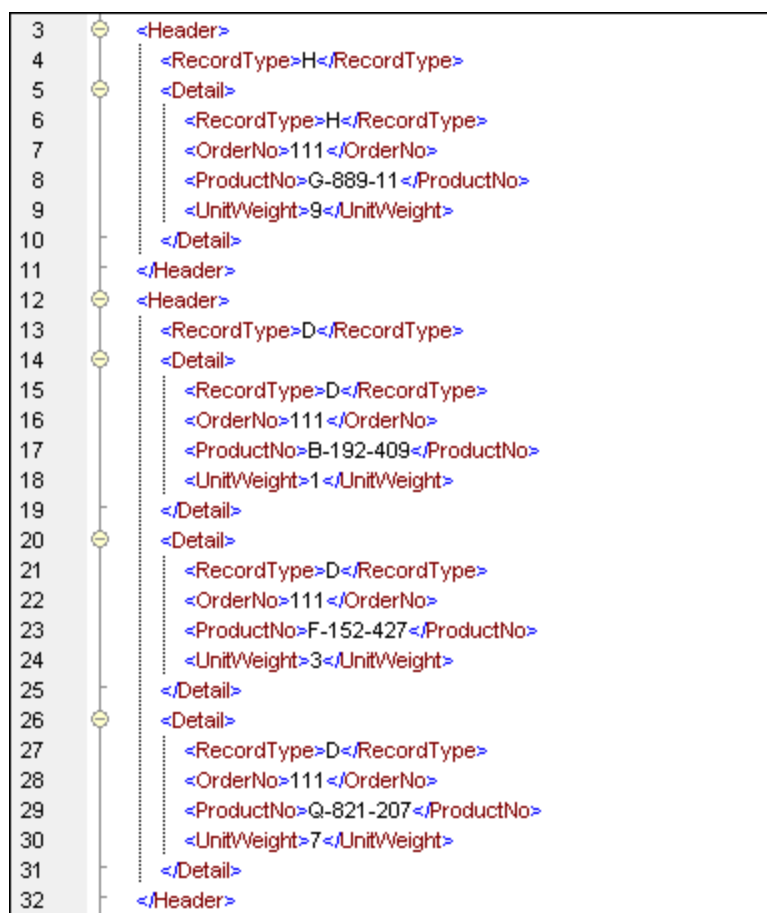
nebeneinanderliegender Datenelemente/Nodes in eine neue separate Gruppe platziert wird.



Wenn wir uns die unten gezeigte CSV-Datei ansehen, möchten wir als Ergebnis alle Header und Detail-Datensätze in einer eigenen Gruppe haben.

Head/Detail	OrderNo	ProdNo	Unitweight	Field5	Field6	Field7	Field8	Fi
string	string	string	string	string	string	string	string	st
H	111	G-889-11	9		Container ship			
D	111	B-192-409	1	2	232	Barley		
D	111	F-152-427	3	1	456	Corn		
D	111	Q-821-207	7	5	52	Coconut		
H	222	A-978-4	22		Air freight			
D	222	M-623-111	8	8	78	Oil		
D	222	L-524-201	2	3	669	Miscellaneous		

- Eine neue Gruppe mit dem ersten Element, in diesem Fall H, wird begonnen.
- Da das nächste Element (oder der Schlüssel) in der Sequenz unterschiedlich ist, d.h. D, wird eine zweite Gruppe namens D begonnen.
- Die nächsten beiden D-Elemente werden nun zur selben Gruppe D hinzugefügt, da sie vom selben Typ sind.
- Eine neue Gruppe H mit einem einzigen H-Element wird begonnen.
- Diese wird gefolgt von einer neuen D-Gruppe mit zwei D-Elementen.

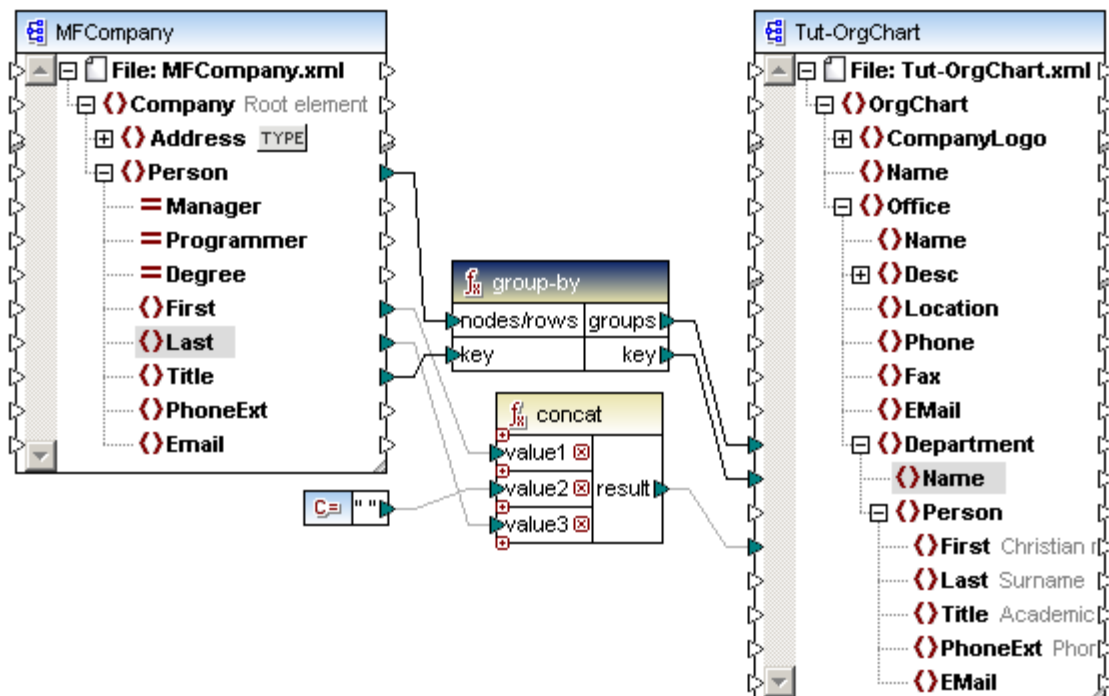


Beachten Sie, dass die Funktion "group-adjacent" als Gruppierungsschlüssel den **Inhalt** des Node/Datenelements verwendet! Die Datensätze werden in der Zieldatei anhand des Inhalts des Felds "Head-Detail" nach Datentyp gruppiert.

group-by

Gruppiert die Input-Sequenz nach eindeutigen Schlüsseln und gibt die verschiedenen Gruppen zusammen mit ihren Schlüsseln aus. Ein Beispiel dafür sehen Sie unten:

- Der Schlüssel, der die einzelnen Gruppen der Quellkomponente definiert, ist das Datenelement **Title**. Anhand dieses Elements werden die Personen des Unternehmens gruppiert.
- Der Gruppenname wird in das Datenelement "Department/Name" der Zielkomponente platziert, wobei der miteinander verkettete Vor- und Nachname (First, Last) in den Sub-Eintrag (Person/First) platziert wird.



Beachten Sie, dass die Funktion "group-by" den **Inhalt** des Node/Datenelements als Gruppierungsschlüssel verwendet! Die Personen werden anhand des Inhalts des Felds "Title" gruppiert und anhand dessen auf das Datenelement "Department/Name" in der Zielkomponente gemappt.

Beachten Sie außerdem: Im inkludierten Beispiel sehen Sie einen **impliziten Filter** der Zeilen vom Quelldokument zum Zieldokument. Im Zieldokument hat jedes Department-Element nur die Person-Elemente, die mit dem Gruppierungsschlüssel (key) **übereinstimmen**, wenn die entsprechende Hierarchie on-the-fly von der "group-by"-Komponente erstellt wird.

Wenn Sie eine flache Hierarchie mit einem dynamischen Output-Dateinamen haben, die zum Teil anhand des key-Werts erstellt wird, so bleibt der implizite Filter weiterhin vorhanden. Dadurch müssen sie die 'groups'-Ausgabe möglicherweise nicht mit einem Element in der Zielkomponente verbinden.

Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie das Ergebnis des Mappings.


```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3  <Office>
4  <Department>
5  <Name>Office Manager</Name>
6  <Person>
7  <First>Vernon Callaby</First>
8  <First>Steve Meier</First>
9  </Person>
10 </Department>
11 <Department>
12 <Name>Accounts Receivable</Name>
13 <Person>
14 <First>Frank Further</First>
15 <First>Theo Bone</First>
16 </Person>
17 </Department>

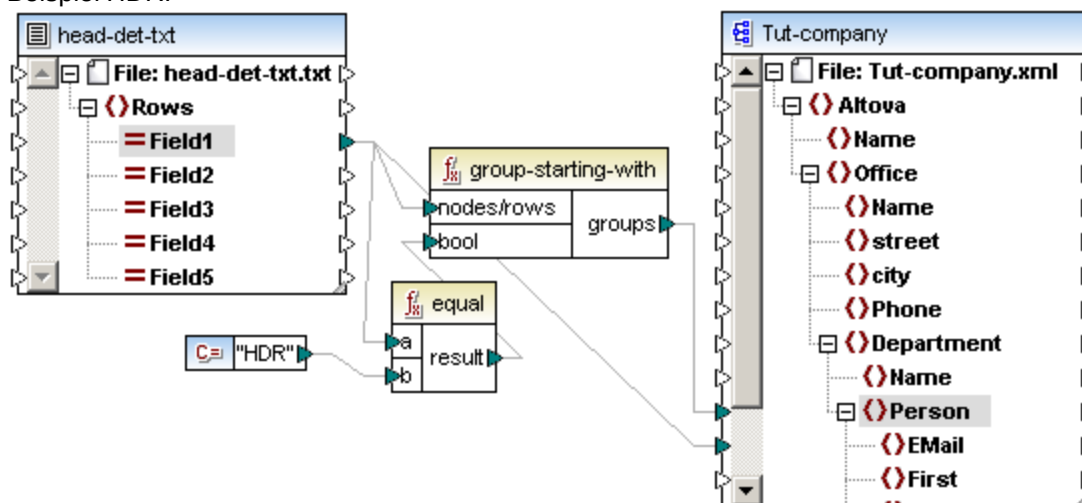
```

group-starting-with

Diese Funktion gruppiert die Input-Sequenz nach dem bereitgestellten Datenelement, wenn es in den Quelldaten vorhanden ist. Zum Testen der Input-Daten wird eine Boolesche Funktion verwendet.

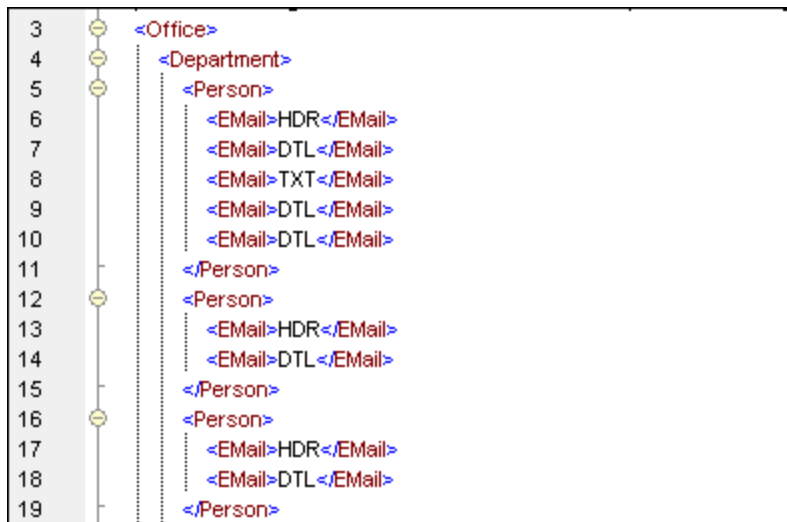
Field1	Field2	Field3	Field4	Field5
string	string	string	string	string
HDR	custid0001	ordid	0001	EUR
DTL	itemABC	qty0100	price	0001.100
TXT	please	deliver	ASAP	
DTL	itemxx2	qty0001	price	0010.000
DTL	itemDDD	qty0010	price	0010.500
HDR	custid0002	ordid	0001	EUR
DTL	itemABC	qty0100	price0001.100	
HDR	custid0003	ordid	0002	USD
DTL	itemDEF	qty0003	price	200.000

Die Funktion erstellt Gruppen auf Basis des **ersten** Datenelements einer Gruppe, in diesem Beispiel HDR.



Der **Wert** der Datenelemente/Nodes muss nicht identisch oder überhaupt vorhanden sein. Das

Node "**Pattern**" (Muster), d.h. die Namen der Nodes/Datenelemente müssen identisch sein, damit die Gruppierung erfolgen kann.

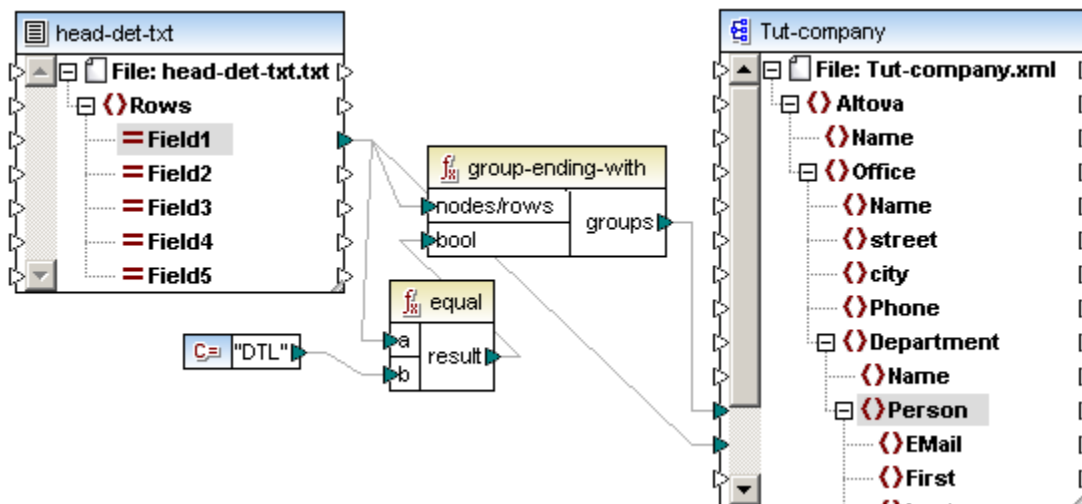


Im Ergebnis oben sehen Sie, dass für jedes HDR-Element eine neue Gruppe begonnen wurde.

group-ending-with

Diese Funktion ist die Ergänzung zur "group-starting-with" Funktion und beendet jede Gruppe der Input-Sequenz anhand des gelieferten Datenelements, falls dieses in der Quelldatei vorhanden ist. Zum Testen der Input-Daten wird eine Boolesche Funktion verwendet.

In diesem Beispiel wird dieselbe Quellkomponente wie im "group-starting-with"-Beispiel oben verwendet. Sie sehen das Ergebnis, wenn DTL als das "group-ending-with"-Datenelement verwendet wird.



In diesem Fall muss der **Wert** des Datenelements/Node nicht identisch oder überhaupt vorhanden sein. Das Node "**Pattern**" (Muster), d.h. die Namen der Nodes/Datenelemente müssen identisch sein, damit die Gruppierung erfolgen kann.



Im Ergebnis oben sehen Sie, dass, immer wenn DTL das letzte Element sein kann, eine neue Gruppe begonnen wurde.

set-empty

Dient zum Generieren einer leeren Sequenz für einen bestimmten Node. Wenn diese Funktion mit einem übergeordneten Node verbunden ist, werden auch alle untergeordneten Nodes auf leer gesetzt. Dadurch wird der Node praktisch aus der Zielkomponente entfernt.

Damit können Sie [Standardwerte](#) eines XBRL-Dokuments, die weiter oben in der XBRL-Komponente/Taxonomie definiert wurden.

13.18 Verwendung von DTDs als "Schema-Komponenten"

Versionen ab MapForce 2006 SP2 unterstützen Namespace-fähige DTDs für Quell- und Zielkomponenten. Die Namespace URIs werden aus den DTD "xmlns"-Attributdeklarationen extrahiert, um Mappings zu ermöglichen.

Hinzufügen von DTD Namespace URIs

Es gibt allerdings einige DTDs, z.B. DTDs, die von StyleVision verwendet werden, die xmlns*-Attributdeklarationen ohne Namespace URIs verwenden. Diese DTDs müssen erweitert werden, um in MapForce verwendet werden zu können:

- Die DTD muss durch Definition des xmlns-Attributs mit der Namespace URI geändert werden, siehe unten:

```
<!ATTLIST fo:root
  xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
  ...
>
```

13.19 Überprüfung der Typkonvertierung

Ab der Version MapForce 2012 **SP2** führen die generierten Applikationen und die Vorschau (im Built-In-Ausführungsprozessor) entsprechend den XSLT2- und XQUERY-Vorgaben eine genauere Überprüfung des Ergebnisses auf Typkonvertierungsfehler durch.

Die Konvertierung der Werte von einem Typ in einen anderen kann nun zu einem Runtime-Fehler führen, wohingegen in früheren Versionen von MapForce ein Ergebnis erzeugt wurde.

Beispiel: Konvertierung von xs:string 'Hello', in xs:decimal

MapForce 2012 Versionen bis einschließlich **SP1**:

XSLT:	'Hello' (oder 'NaN' bei Übergabe an eine Funktion, die mit Zahlen arbeitet)
XSLT2:	Fehler: "ungültiger lexikalischer Wert"
Xquery:	Fehler: "ungültiger lexikalischer Wert"
Vorschau mit Built-In-Ausführungsprozessor:	0
C++ app:	0
C# app:	Fehler: "Werte können nicht konvertiert werden"
Java app:	Fehler: "Werte können nicht konvertiert werden"

MapForce 2012 **SP2**:

XSLT:	'Hello' (oder 'NaN' bei Übergabe an eine Funktion, die mit Zahlen arbeitet)
XSLT2:	Fehler: "ungültiger lexikalischer Wert"
Xquery:	Fehler: "ungültiger lexikalischer Wert"
Vorschau mit dem Built-In-Ausführungsprozessor:	Fehler: "String-Wert 'Hello' konnte nicht in Dezimalwert konvertiert werden"
C++ app:	Fehler: "Werte können nicht konvertiert werden"
C# app:	Fehler: "Werte können nicht konvertiert werden"
Java app:	Fehler: "Werte können nicht konvertiert werden"

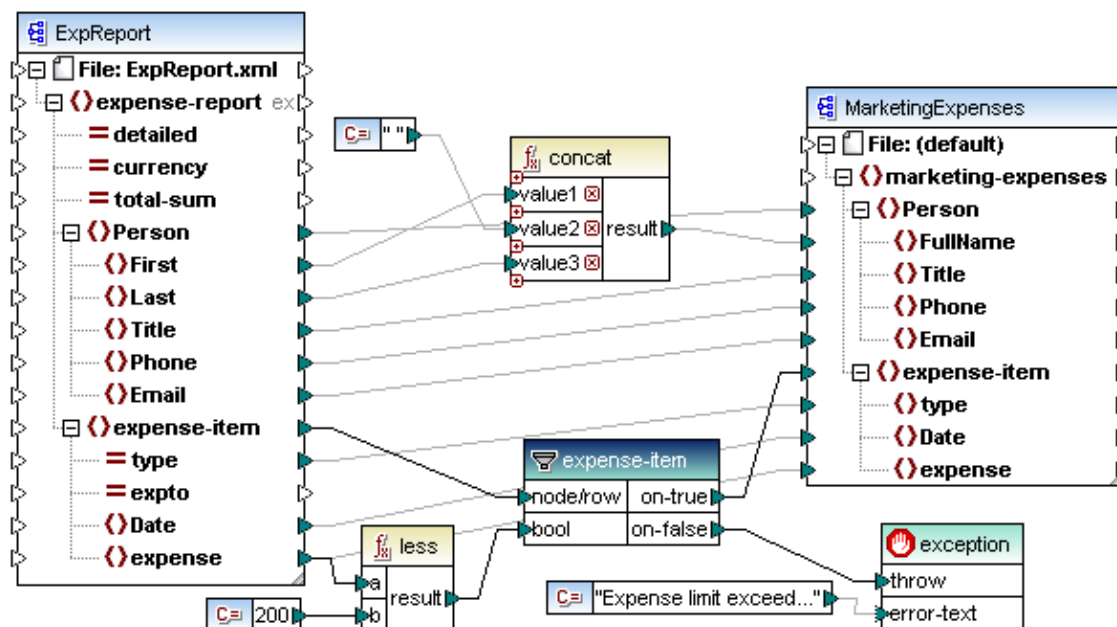
Bei Auftreten von Typkonvertierungsfehlern überprüfen Sie, ob die Typen korrekt behandelt wurden. Verwenden Sie z.B. die Funktion lang:numeric(), um zu überprüfen, ob der Quellwert in eine Zahl konvertiert werden kann, und verwenden Sie anschließend eine if-else-Komponente, um einen anderen Wert zu übergeben, falls dies fehlschlägt (z.B. eine Konstante, die im Parameter "value-false" den Wert -1 enthält).

13.20 Ausnahmeereignisse

MapForce bietet Unterstützung für die Definition von Ausnahmeereignissen. Sie können eine Bedingung definieren, die einen Fehler auslöst. Wenn die Bedingung erfüllt wird, wird eine benutzerdefinierte Meldung angezeigt und das Mapping wird gestoppt. Die Datei **ExpenseLimit.mfd** im Ordner **...MapForceExamples** ist ein Beispiel-Mapping, das eine Ausnahmeereignis-Funktion enthält.

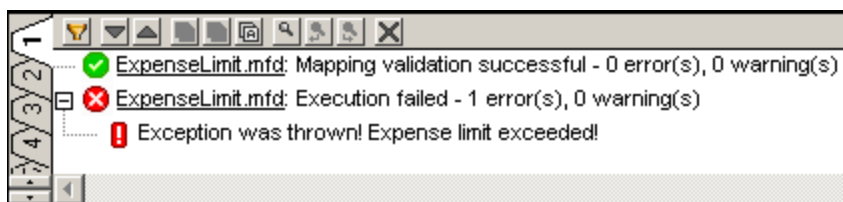
So fügen Sie eine Ausnahmeereignis-Komponente ein:

- Wählen Sie die Menüoption **Einfügen | Ausnahmeereignis**, oder klicken Sie auf das Symbol "Ausnahmeereignis"  in der Symbolleiste.



Im obigen Beispiel sehen Sie, wie Ausnahmeereignisse in Mappings definiert werden. Das Ausnahmeereignis soll ausgelöst werden, wenn das Ausgabenlimit überstiegen wird, d.h. wenn "expense" in diesem Beispiel höher als 200 ist.

- Die **less**-Komponente überprüft, ob die Ausgaben (expense) weniger als 200 betragen, wobei das Boolesche Ergebnis an die Filter-Komponente übergeben wird.
- Wenn das Boolesche Ergebnis **false** ist, d.h. wenn die Ausgaben größer oder gleich 200 sind, wird durch den **on-false** Parameter das Ausnahmeereignis aktiviert und das Mapping wird gestoppt. (Beachten Sie, dass Sie das Ausnahmeereignis auch mit dem on-true-Parameter verbinden können, wenn dies das gewünschte Ergebnis ist.)
- Der Fehlertext (Expense limit exceede) aus der Konstanten-Komponente wird auf den Parameter **error-text** des Ausnahmeereignisses gemappt.
- Der Fehlertext wird auf dem Register "Ausgabe" angezeigt und auch beim Ausführen des kompilierten Codes.



Bitte beachten Sie:

Die Platzierung der filter-Komponente in diesem Beispiel ist ungemein wichtig:

- Es müssen **beide Parameter** der filter-Komponente, on-true und on-false, gemappt werden! Einer der beiden muss auf die **exception**-Komponente gemappt werden und der andere auf die Zielkomponente, an die die gefilterten Quelldaten übergeben werden. Ist dies nicht der Fall, so wird die Fehlerkomponente nie ausgelöst.
- Die **exception**-Komponente und die **Zielkomponente** müssen **direkt** mit der **filter**-Komponente verbunden werden. Es dürfen keine Funktionen oder anderen Komponenten zwischen den Filter und das Ausnahmeereignis oder die Zielkomponenten platziert werden.
- Bei Generierung von **XSLT 2.0**- und **XQuery**-Code wird im Fenster "Meldungen" eine Meldung angezeigt, dass die Ausführung fehlgeschlagen ist. Das entsprechende XSLT2 bzw. XQuery-Register wird angezeigt, und die Zeile, die das Ausnahmeereignis ausgelöst hat, erscheint markiert.

13.21 Katalogdateien in MapForce

MapForce unterstützt einen Teil der OASIS XML-Kataloge. Mit Hilfe von Katalogen kann MapForce 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 in MapForce funktioniert, wie unten beschrieben.

RootCatalog.xml

MapForce lädt eine Datei namens `RootCatalog.xml` (*Struktur siehe unten*), 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.

```
<?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>
```

Die in `RootCatalog.xml` inkludierten Katalogdateien werden durchsucht und die URIs werden entsprechend den Zuordnungen in den Katalogdateien aufgelöst. Beachten Sie, dass die Katalogdateien `catalog.xml` in den Ordnern `Schemas` und `XBRL` des durch die Variable `%AltovaCommonFolder%` definierten Ordners durchsucht werden. (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 WSDL) und XBRL-Taxonomien auf URIs, die auf lokal gespeicherte Kopien der entsprechenden Schemas verweisen. Diese Schemas werden bei der Installation von MapForce im Altova Ordner "Common" installiert. Achten Sie darauf, 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 MapForce Applikationsordner installiert. Die Datei `CustomCatalog.xml` befindet sich im Ordner `MyDocuments/Altova/MapForce`. Die `Catalog.xml` Dateien befinden sich in den jeweiligen Schemaordnern. Diese Schemaordner befinden sich im Ordner:
`%AltovaCommonFolder%\Schemas` and `%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\CommonMapForce
<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 MapForce 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 MapForce unterstützte Katalogteilmenge

Wenn Sie Einträge für `CustomCatalog.xml` (oder jede andere Katalogdatei, die von MapForce 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 `rewriteSystem` 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 MapForce 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/xhtml1/xhtml1-transitional.dtd"/>
```

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

XML-Schema und Kataloge

XML-Schemainformationen sind in MapForce 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 MapForce zu Bearbeitungszwecken Informationen bereitzustellen, falls Sie Dateien gemäß diesen älteren Empfehlungen erstellen wollen.

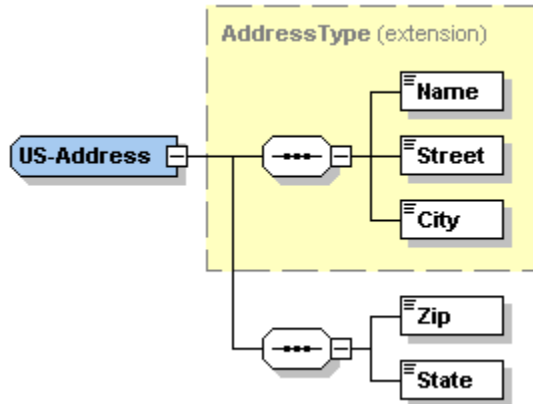
Weitere Informationen

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

13.22 Derived XML-Schema-Typen - Mappen auf

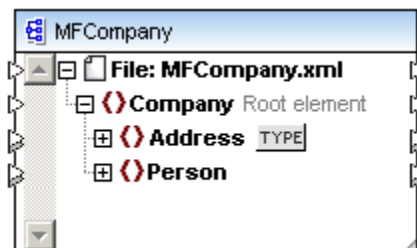
MapForce unterstützt das Mappen von/auf Derived Types eines complexType. Derived Types sind complexTypes eines XML-Schemas, die das Attribut `xsi:type` verwenden, um die spezifischen Derived Types zu identifizieren.

In der Abbildung unten sehen Sie die Definition des Derived Type "US-Address" in XMLSpy. Der Basistyp (oder der ursprüngliche complexType) ist in diesem Fall **AddressType**. Es wurden zwei zusätzliche Elemente hinzugefügt, um den Derived Type US-Address zu erstellen.

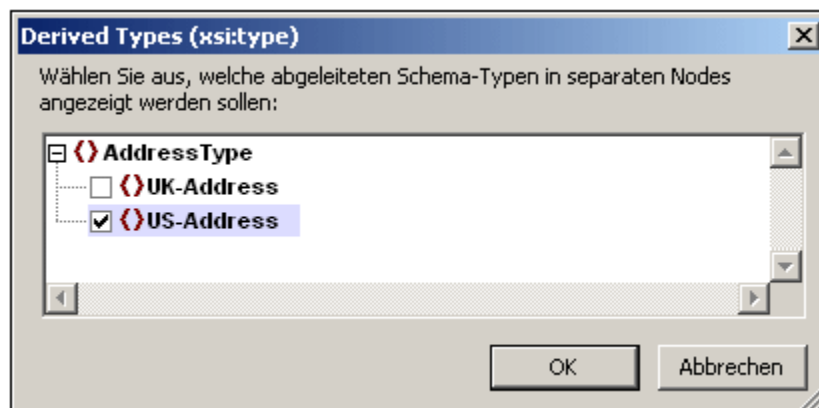


Mappen auf Derived Types in MapForce:

1. Fügen Sie das XML-Schema MFCompany.xsd ein, das im Ordner ...Tutorial zur Verfügung steht, klicken Sie auf "Überspringen" und wählen Sie anschließend als Root-Element "Company" aus.



2. Klicken Sie auf die Schaltfläche TYPE rechts vom Element "Address", welche anzeigt, dass in der Schemakomponente Derived Types vorhanden sind.

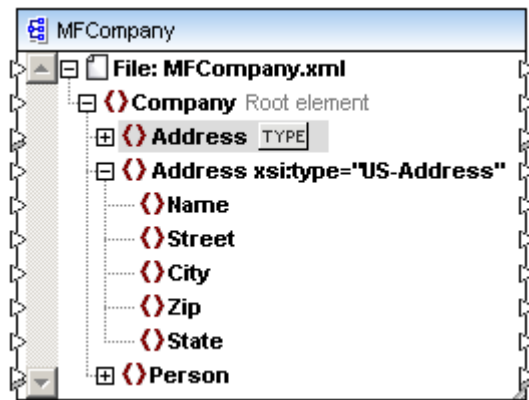


3. Aktivieren Sie das Kontrollkästchen neben dem Derived Typen, den Sie verwenden

möchten, z.B. US-Address und bestätigen Sie mit OK.

Ein neues Element **Address xsi:type="US-Address"** wurde zur Komponente hinzugefügt.

4. Klicken Sie auf die Erweiterungsschaltfläche, um die mapbaren Datenelemente des Elements zu sehen.



5. Sie können nun direkt von/auf diese Datenelemente mappen.

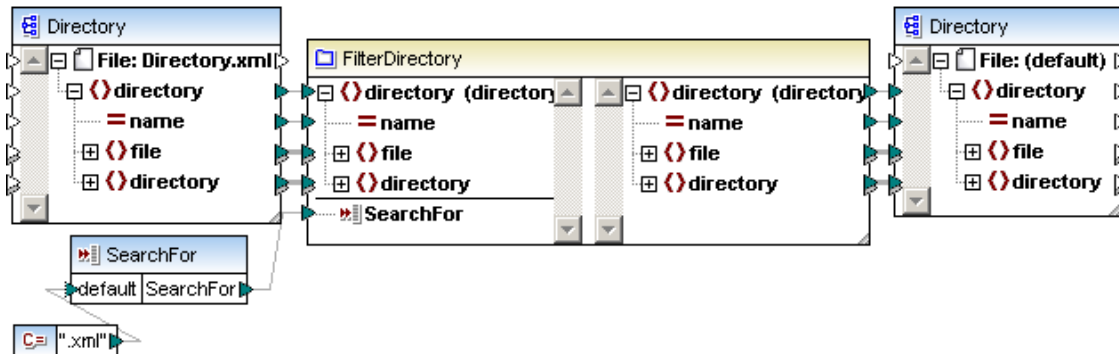
Bitte beachten Sie:

Sie können mehrere Derived Types inkludieren/einfügen. Wählen Sie diese dazu im Dialogfeld "Derived Types" aus. Jedes davon hat in der Komponente ein eigenes xsi:type-Element.

13.23 Rekursives benutzerdefiniertes Mapping

In diesem Abschnitt wird beschrieben, wie das Mapping **RecursiveDirectoryFilter.mfd** aus dem Ordner ...\\MapForceExamples erstellt wurde und wie rekursive Mappings erstellt werden. Der Projektordner MapForceExamples enthält weitere Beispiele für rekursive Mappings.

In der Abbildung unten sehen Sie das fertige Mapping, das die rekursive benutzerdefinierte Funktion FilterDirectory enthält, deren Ziel es ist, in der Quelldatei eine Liste der .xml-Dateien zu filtern.



Die **Quelldatei**, die die Datei- und Verzeichnisdaten für dieses Mapping enthält, ist Directory.xml. Diese XML-Datei liefert, wie unten gezeigt, das Verzeichnis und die Dateidaten in hierarchischer Form.

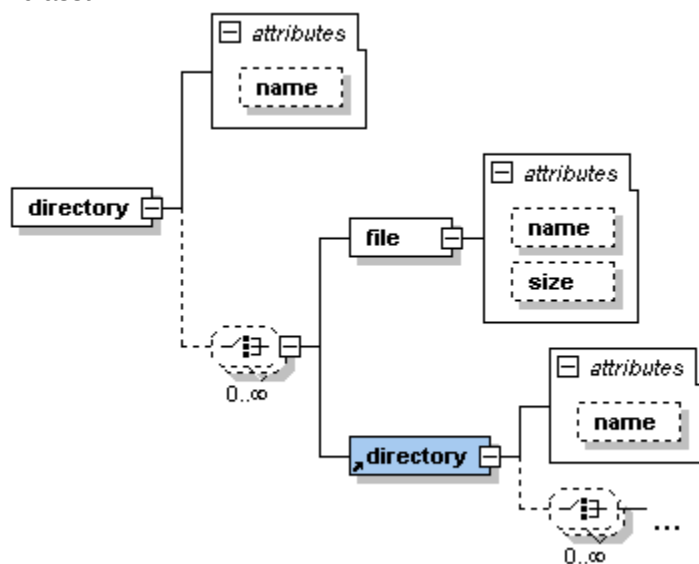
```

24      <directory name="output">
25          <file name="examplesite1.css" size="3174"/>
26          <directory name="images">
27              <file name="blank.gif" size="88"/>
28              <file name="block_file.gif" size="13179"/>
29              <file name="block_schema.gif" size="9211"/>
30              <file name="nav_file.gif" size="60868"/>
31              <file name="nav_schema.gif" size="6002"/>
32          </directory>
33      </directory>
34  </directory>
35      <directory name="Import">
36          <file name="altova.mdb" size="266240"/>
37          <file name="Data_shape.mdb" size="225280"/>
38      </directory>
39      <directory name="IndustryStandards">
40          <directory name="News">
41              <file name="high-tide.jpg" size="10793"/>
42              <file name="Newsml-example.xml" size="5004"/>
43              <file name="nitf-example.xml" size="9327"/>
44          </directory>

```

Die von Directory.xml referenzierte XML-Schemadatei hat ein **rekursives** Element namens "directory", das beliebig viele Unterverzeichnisse und Dateien unterhalb des directory-Elements

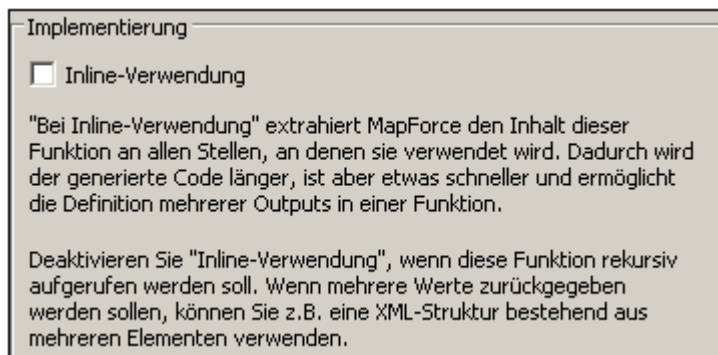
zulässt.



13.23.1 Definieren eines rekursiven benutzerdefinierten Mappings

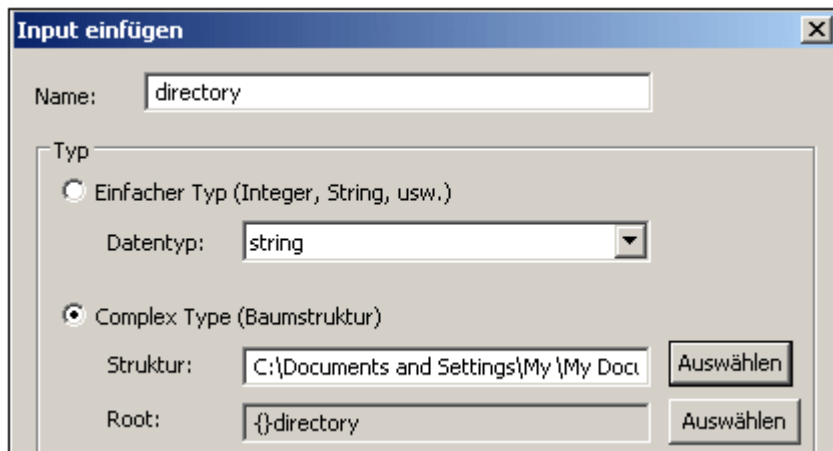
Aktionen im Hauptmappingfenster:

1. Wählen Sie den Befehl **Funktion | Benutzerdefinierte Funktion erstellen**, um mit der Erstellung der Funktion zu beginnen und geben Sie einen Namen ein, z.B. FilterDirectory.
2. Stellen Sie sicher, dass Sie das Kontrollkästchen **Inline-Verwendung** in der Gruppe **Implementierung** deaktiviert haben, um die Funktion rekursiv zu machen und klicken Sie anschließend auf OK.

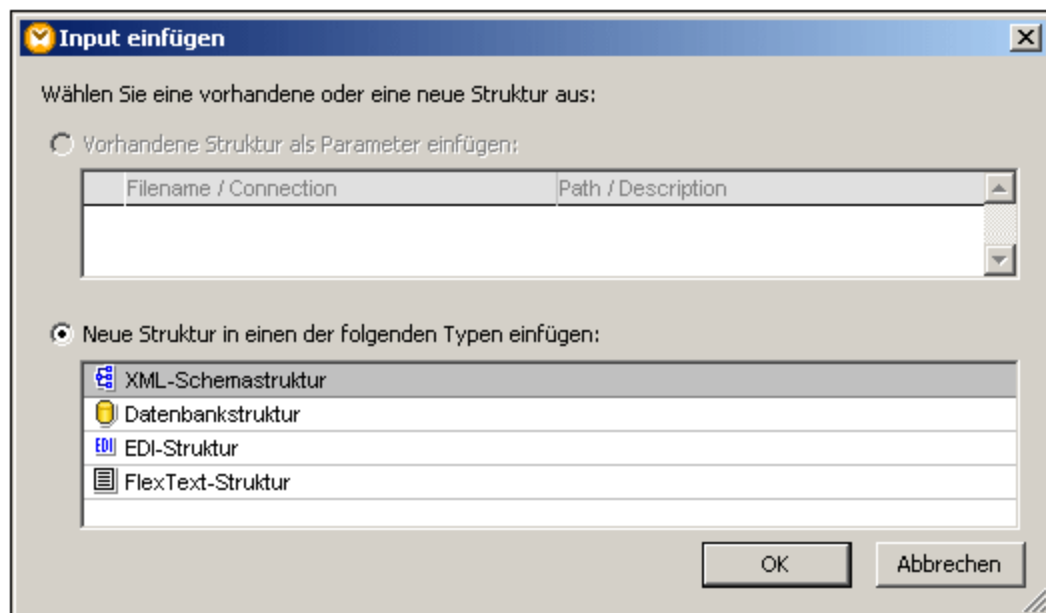


Sie befinden sich jetzt im Fenster **FilterDirectory**, wo Sie die benutzerdefinierte Funktion erstellen können.

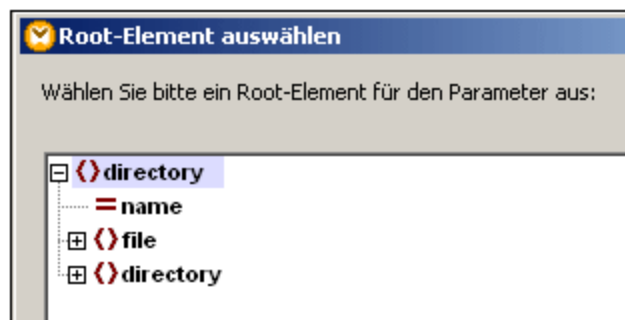
3. Wählen Sie **Funktion | Input-Komponente einfügen** um eine Input-Komponente einzufügen.
4. Geben Sie der Komponente einen Namen, z.B. directory und klicken Sie auf das Optionsfeld **Complex Type** (Baumstruktur).



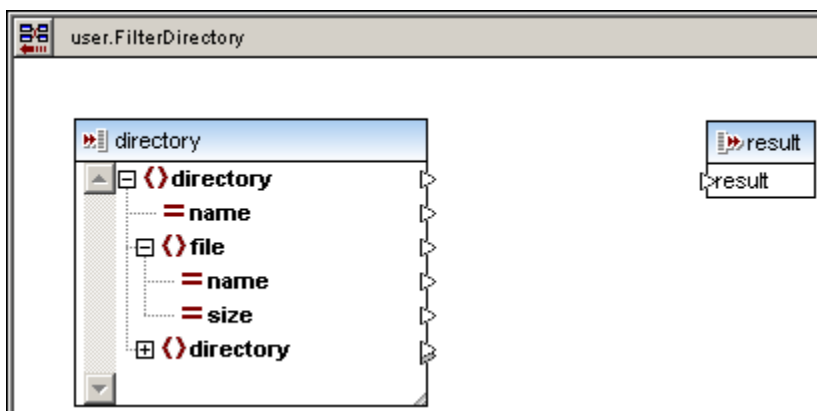
5. Klicken Sie auf die Schaltfläche **Auswählen** und wählen Sie die Datei directory.xsd im Verzeichnis ...\\MapForceExamples.



6. Wählen Sie in die Datei **Directory.xsd** im Ordner ...MapForceExamples aus und klicken Sie auf die Schaltfläche "Öffnen".
7. Klicken Sie nochmals auf OK, wenn Sie gebeten werden, das Root-Element auszuwählen, welches "directory" sein sollte.

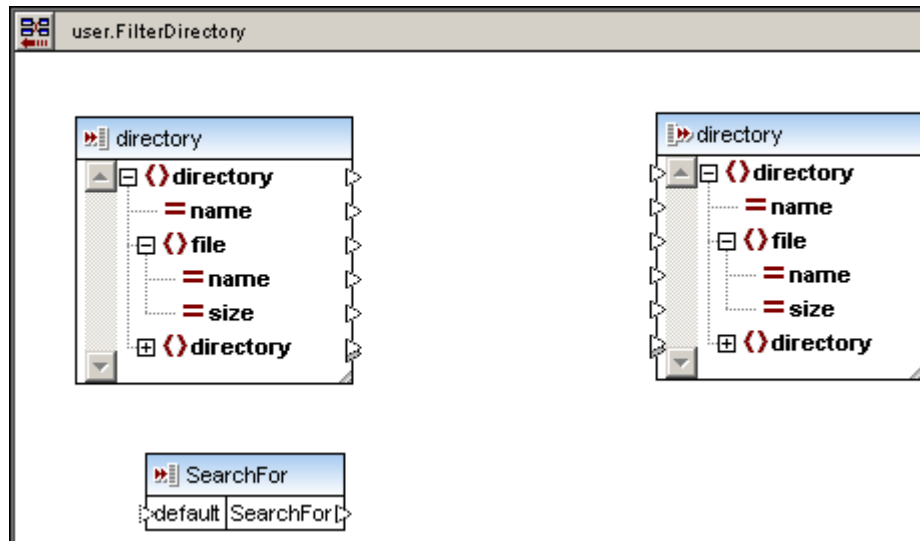


8. Klicken Sie wieder auf Ok, um den complex Input-Parameter einzufügen. Unten sehen Sie die benutzerdefinierte Funktion.



9. Löschen Sie die simple-Output-Komponente "result", da hier eine complex Output-Komponente eingefügt werden muss.

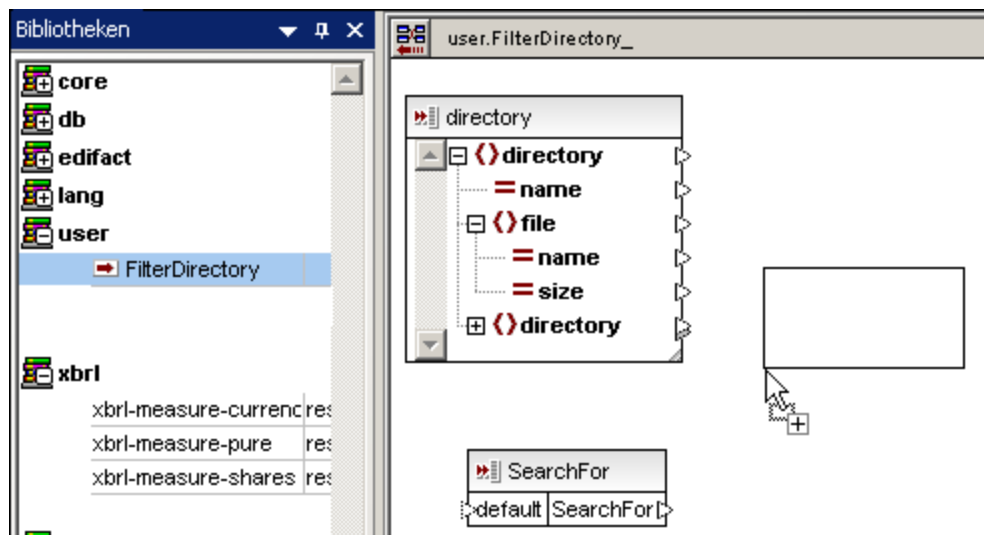
10. Wählen Sie **Funktion | Output-Komponente einfügen...**, um eine Output-Komponente einzufügen und verwenden Sie dieselbe Methode wie oben, um die Output-Komponente "directory" zu einem complexType zu machen. Sie haben jetzt zwei complex-Komponenten, eine Input- und eine Output-Komponente.
11. Wählen Sie **Funktion | Input-Komponente einfügen...** und fügen Sie eine Komponente vom Typ Simple type ein und geben Sie einen Namen ein, z.B. **SearchFor**. Deaktivieren Sie das Kontrollkästchen "Input ist erforderlich".



Einfügen der rekursiven benutzerdefinierten Funktion

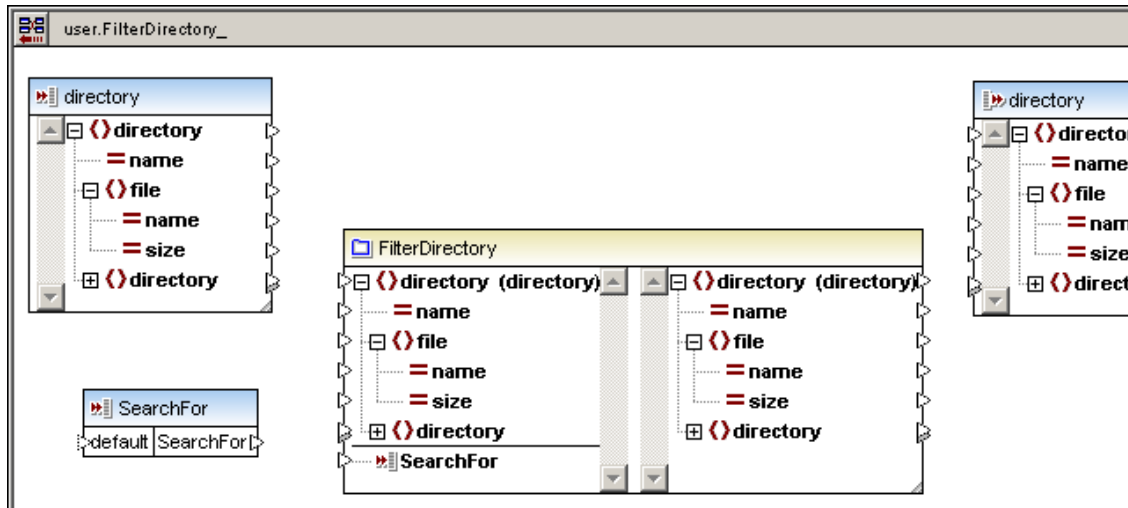
Wir haben jetzt alle erforderlichen Input- und Output-Komponenten für die benutzerdefinierte Funktion definiert. Nun muss die "unfertige" Funktion in das aktuelle Fenster für die benutzerdefinierte Funktion eingefügt werden. (Sie können dies jedoch zu jedem beliebigen Zeitpunkt tun.)

1. Suchen Sie die Funktion FilterDirectory in Fenster **Bibliotheken** im Abschnitt **user**.
2. Klicken Sie auf FilterDirectory und ziehen Sie sie in das Fenster FilterDirectory, in dem Sie gerade gearbeitet haben.

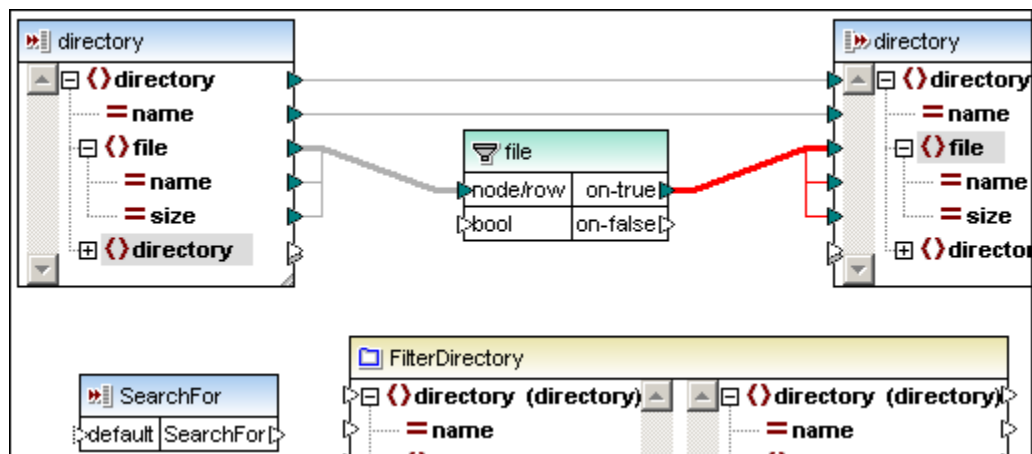


Java ausgewählt

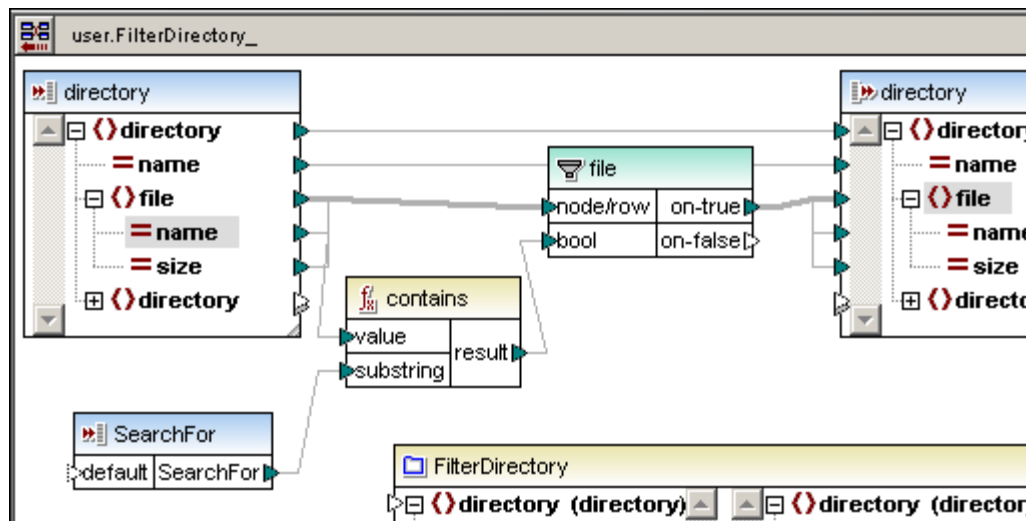
Die benutzerdefinierte Funktion wird nun als rekursive Komponente im Fenster für die benutzerdefinierte Funktion angezeigt.



3. Verbinden Sie die Datenelemente **directory**, **name** und **file** der Input-Komponente mit den entsprechenden Datenelementen in der Output-Komponente.
4. Klicken Sie mit der rechten Maustaste auf den Konnektor zwischen den Datenelementen **file** und wählen Sie den Befehl "Filter einfügen", um eine Filterkomponente einzufügen.
5. Klicken Sie mit der rechten Maustaste auf den on-true-Konnektor und wählen Sie im Kontextmenü den Befehl **Alles kopieren**. Die Konnektoren ändern ihr Aussehen und werden als "Alles kopieren"-Konnektoren angezeigt.



6. Fügen Sie aus der Bibliothek **Core | String functions** eine Contains-Funktion ein.
7. Verbinden Sie **name** mit **value** und den Parameter **SearchFor** mit **substring**, anschließend **result** mit dem Bool-Element des Filters.



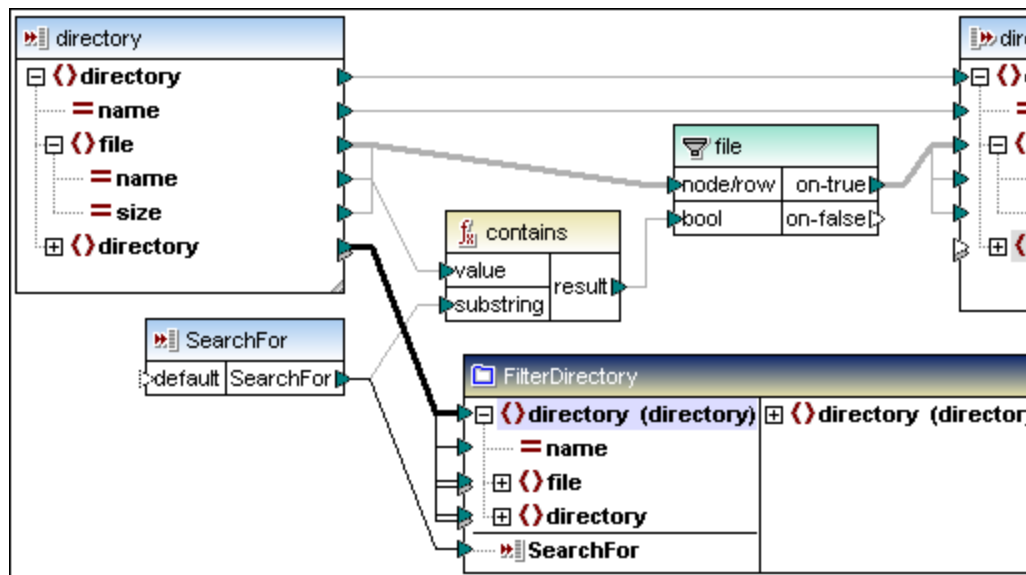
8. Verbinden Sie das Datenelement SearchFor der Input-Komponente mit dem Datenelement SearchFor der benutzerdefinierten Funktion.

Definieren der Rekursion

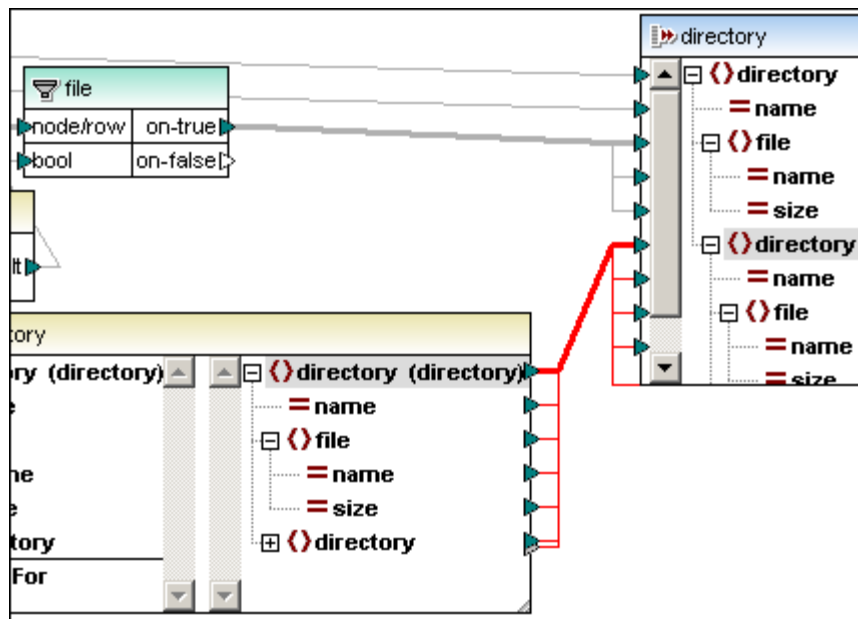
Zu diesem Zeitpunkt ist das Mapping einer einzigen Verzeichnisrekursionsebene fertig. Nun müssen wir nur noch definieren, wie ein Unterverzeichnis verarbeitet werden soll.

Stellen Sie sicher, dass in der Symbolleiste die Schaltfläche "Automatisch verbinden"  aktiv ist:

1. Verbinden Sie das untere **directory** Element der Input-Komponente mit dem oberen **directory** Element der rekursiven benutzerdefinierten Funktion.



2. Verbinden Sie das obere Output directory-Element der benutzerdefinierten Funktion mit dem unteren directory-Element der Output-Komponente.
3. Klicken Sie mit der rechten Maustaste auf den Konnektor, wählen Sie im Kontextmenü den Befehl "Alles kopieren" und klicken Sie auf Ok, wenn Sie gefragt werden, ob Sie eine "Alles-kopieren"-Verbindung erstellen möchten.

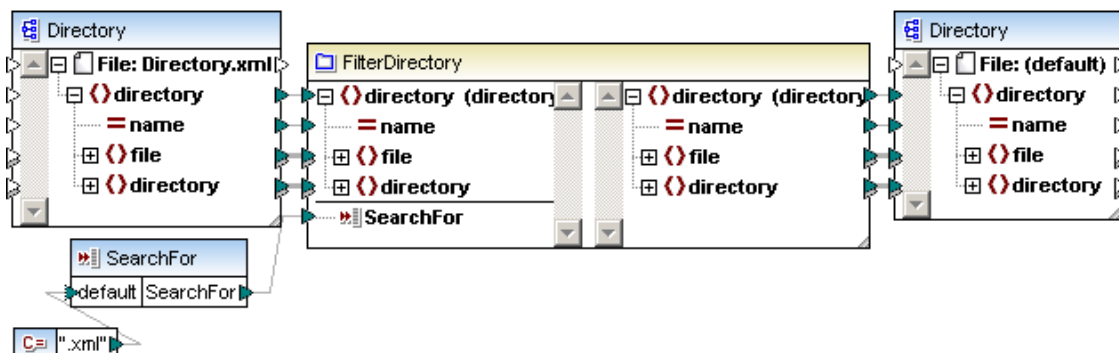


Damit ist die Definition der benutzerdefinierten Funktion in diesem Fenster abgeschlossen.

Klicken Sie auf die Schaltfläche "Zurück zum Hauptmapping-Fenster" , um das Mapping dort fertig definieren zu können.

Hauptmapping-Fenster

1. Ziehen Sie die Funktion FilterDirectory aus dem **user** Abschnitt des Fensters "Bibliotheken" in das **Hauptmapping**.
2. Wählen Sie den Befehl **Einfügen | XML-Schemadatei** um Directory.xsd einzufügen und wählen Sie Directory.xml als Instanzdatei.
3. Fügen Sie auf dieselbe Art Directory.xsd ein und wählen Sie **Überspringen** um die Output-Komponente zu erstellen.
4. Fügen Sie eine Konstantenkomponente ein, anschließend eine Input-Komponente, z.B. SearchFor.
5. Erstellen Sie die Verbindungen wie in der Abbildung unten gezeigt.
6. Wenn Sie auf beiden Seiten der benutzerdefinierten Komponente die Verbindungen auf der obersten Ebene erstellen, directory zu directory, klicken Sie mit der rechten Maustaste auf den Konnektor und wählen Sie im Kontextmenü den Befehl **Alles kopieren**.



7. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

Anmerkungen:

Wenn Sie in der Komponente "Directory" auf das unterste "directory" Datenelement doppelklicken, wird eine neue Rekursionsebene geöffnet, d.h. Sie sehen eine neue **directory | file | directory** Unterebene. Mit Hilfe des "Alles kopieren"-Konnektors werden automatisch alle vorhandenen Rekursionsebenen in der XML-Instanz verwendet und Sie müssen die Rekursionsebenen nicht mehr manuell erweitern.

Kapitel 14

StyleVision Power Stylesheets in der Vorschau

14 StyleVision Power Stylesheets in der Vorschau

MapForce ist nun in der Lage, XML-Schema- und XBRL-Komponenten mit Hilfe eines damit verknüpften StyleVision Power Stylesheet (SPS) in einer Vorschau anzuzeigen/darzustellen, d. h. die Daten der Zielkomponente können in einer Vorschau als HTML-, RTF-, PDF- oder Word 2007+-Dokumente angezeigt werden. Wenn Sie die Enterprise Edition von StyleVision verwenden, werden auch Diagramme in diesen Vorschaudokumenten angezeigt.

Um eine derartige Vorschau anzeigen zu können, muss Altova StyleVision entweder als Standalone Edition oder als Teil des Altova MissionKit auf Ihrem Computer installiert sein.

StyleVision Power Stylesheets werden in StyleVision erstellt und einer Komponente in MapForce zugewiesen. Das Stylesheet kann nicht direkt in MapForce bearbeitet oder geändert werden, doch Sie können ein Stylesheet über MapForce in StyleVision öffnen. In der 64-Bit-Version von MapForce werden die Word 2007+-Ansicht und die RTF-Ansicht als nicht eingebettete Applikationen geöffnet.

Wenn StyleVision installiert ist, erhält die MapForce Registerkartenleiste zusätzliche Register und der Zielkomponente wird eine SPS-Datei zugewiesen.

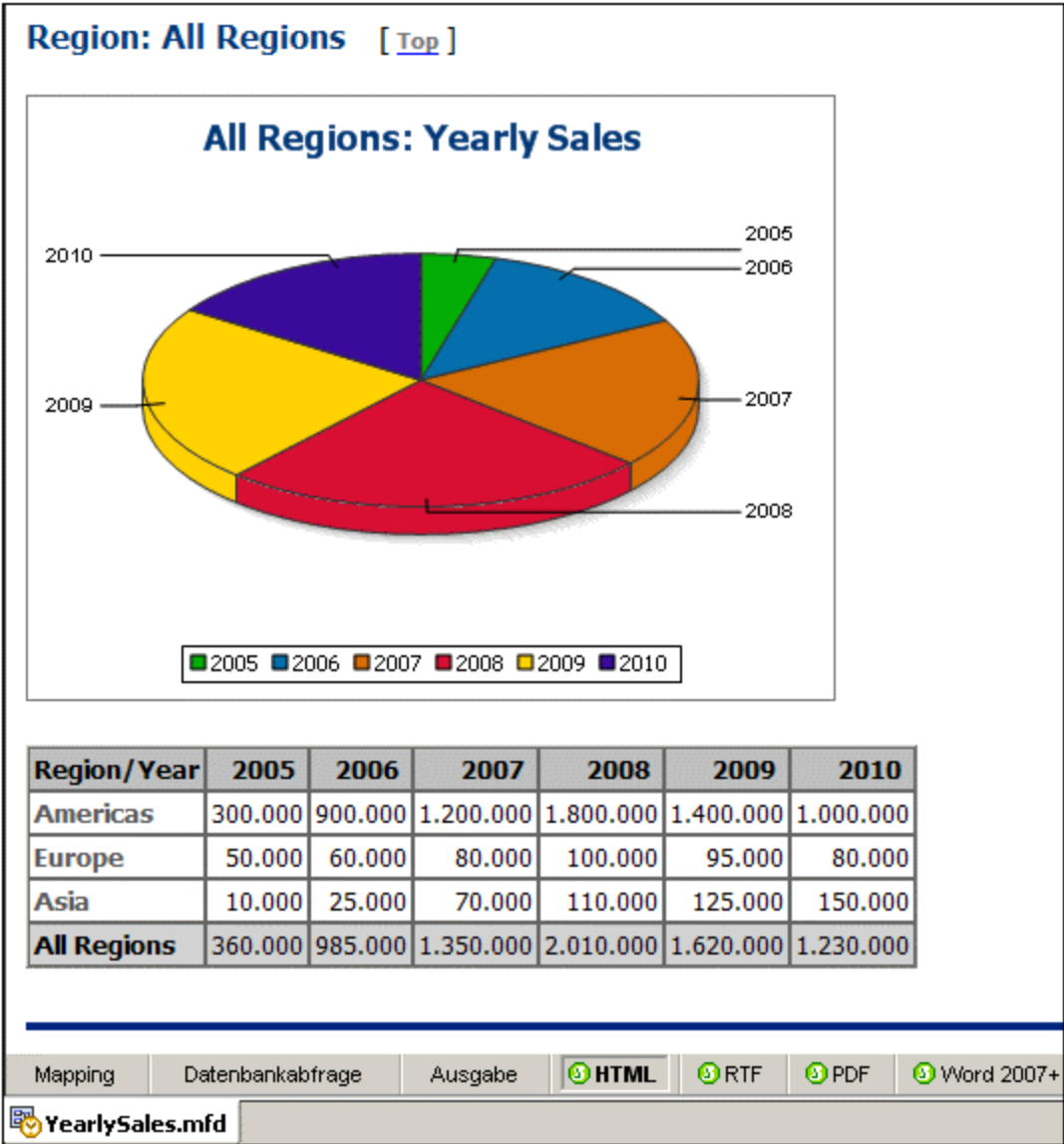
- HTML
- RTF - zeigt eine RTF-Vorschau an
- PDF - dafür wird die FOP-Version 0.93 oder 1.0 benötigt
- Word 2007+ - zeigt eine Vorschau von MS Word-Dokumenten ab Version 2007 und höher an

Bitte beachten Sie:

Welche Register in MapForce zur Verfügung stehen, hängt davon ab, welche Version von StyleVision auf Ihrem Computer installiert ist. Alle oben aufgelisteten Register stehen in der Enterprise Edition zur Verfügung. Wenn diese zusätzlichen Register angezeigt werden, heißt dies, dass der Komponente eine SPS-Datei zugewiesen wurde. Der Ordner ...\\MapForceExamples enthält mehrere Beispiele wo dies der Fall ist.

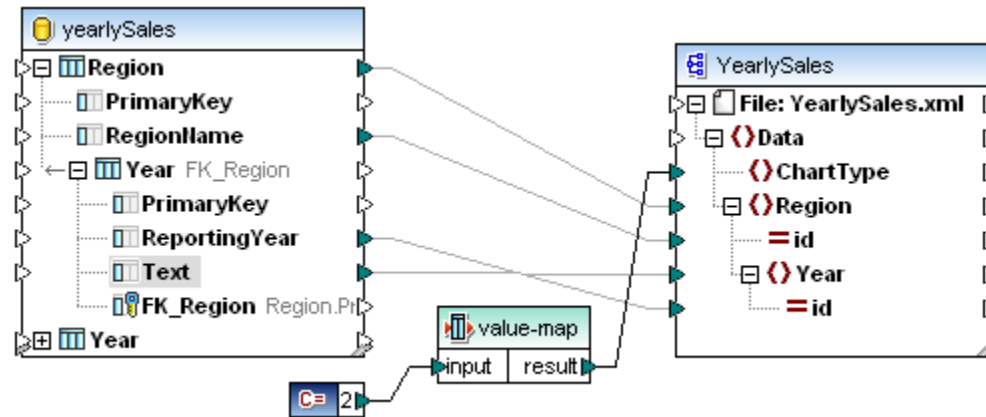
Bei der Professional Edition von StyleVision stehen in MapForce nur die Register "HTML" und "RTF" zur Verfügung.

Anmerkung: Wenn Ihr Mapping mehrere miteinander verknüpfte Komponenten, also ein [verkettetes Mapping](#) enthält, steht die SPS-Vorschau nur für diejenigen Komponenten zur Verfügung, die einen SPS-Eintrag enthalten und bei denen die Schaltfläche "Vorschau"  der jeweiligen Komponente aktiv gesetzt wurde.



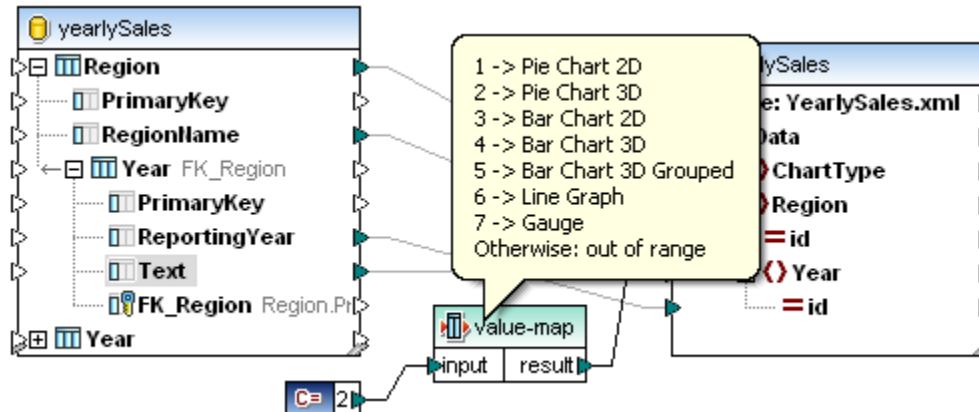
14.1 Zuweisen einer SPS-Datei zu einer Komponente

Sie finden das unten gezeigte Beispiel **YearlySales.mfd** im Ordner ...\\MapForceExamples\\Tutorial.



Durch Doppelklick auf die Konstantenkomponente und Eingabe einer Zahl definieren Sie den Typ des Diagramms, der angezeigt wird, wenn Sie auf das entsprechende StyleVision-Register klicken.

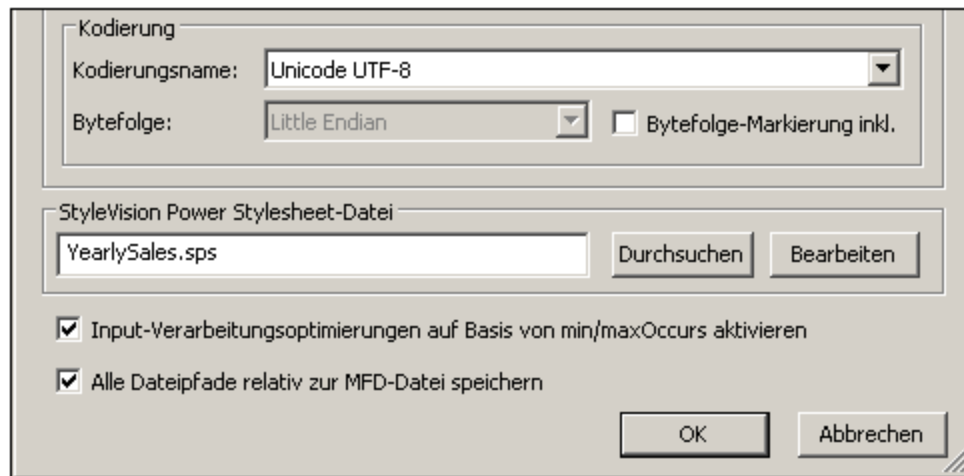
Wenn Sie den Cursor über die value-map Funktion platzieren, werden die verschiedenen Auswahlmöglichkeiten im Popup-Fenster angezeigt, in diesem Fall wird mit dem Wert das Kreisdiagramm 3-D ausgewählt.



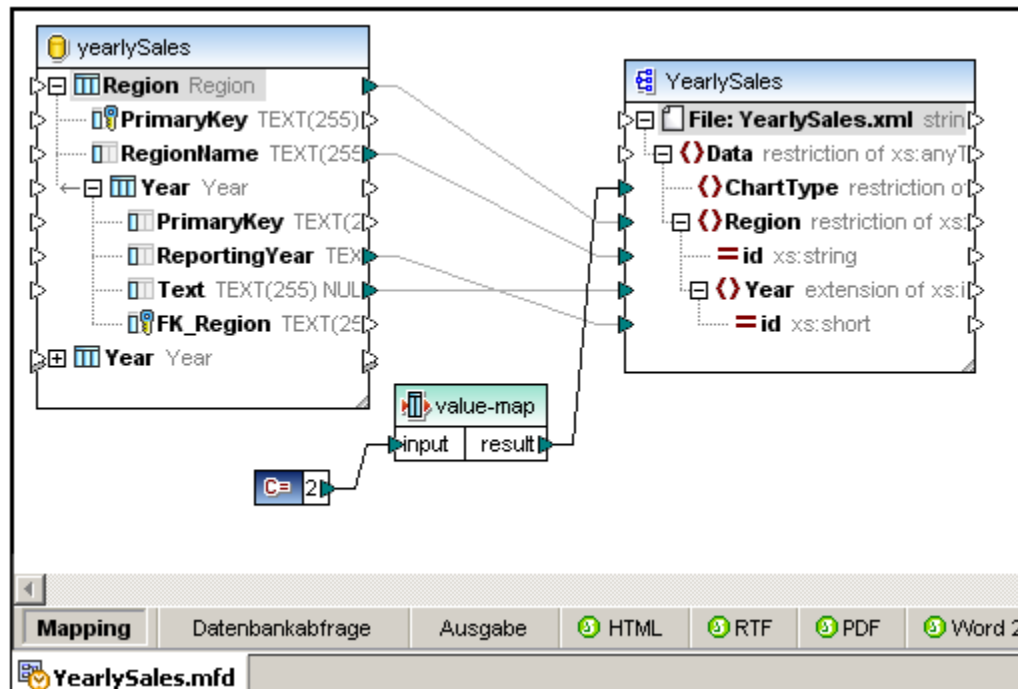
So weisen Sie einer MapForce-Komponente eine SPS-Vorlage zu:

Nachdem Sie eine SPS-Vorlage in StyleVision erstellt oder von jemandem anderen eine Vorlage erhalten haben,

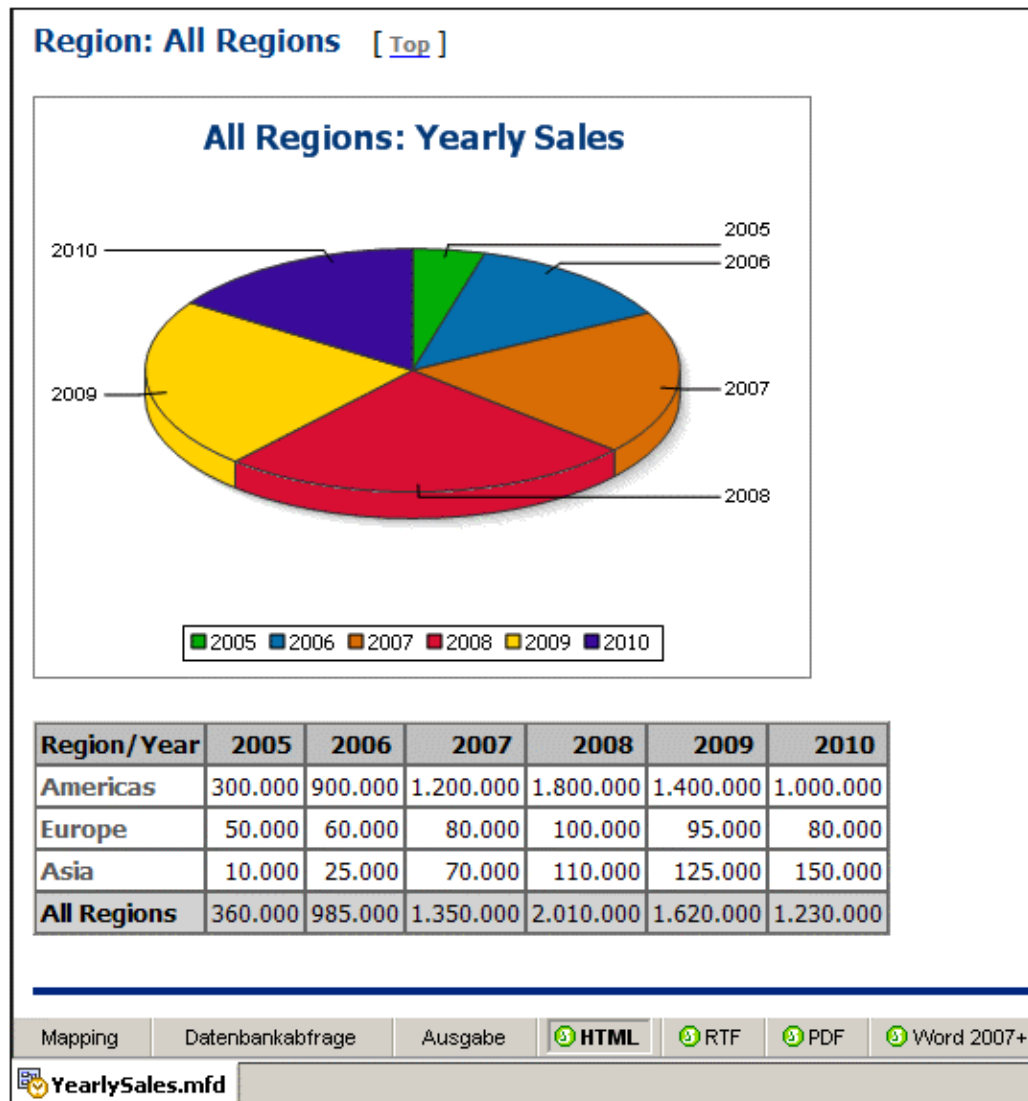
1. Doppelklicken Sie auf die Titelleiste der Komponente, der Sie die SPS-Datei zuweisen möchten, z.B. YearlySales.



2. Klicken Sie auf die Schaltfläche "Durchsuchen", um die StyleVision SPS-Datei auszuwählen und bestätigen Sie die Auswahl mit OK.



Rechts vom Register "Ausgabe" werden die Register HTML, RTF, PDF und Word 2007 + angezeigt. Klicken Sie auf eines dieser Register, um die Komponentendaten im jeweiligen Format darzustellen.



Bitte beachten Sie:

Wenn Sie im Dialogfeld "Komponenteneinstellungen" auf die Schaltfläche "Erstellen..." klicken, werden Sie aufgefordert, den Namen einer SPS-Daten anzugeben und StyleVision wird geöffnet, in dem Sie eine neue SPS-Datei erstellen können. Wenn Sie auf dieselbe Komponente doppelklicken, sobald eine SPS-Datei zugewiesen wurde, wird anstelle der Schaltfläche "Erstellen" die Schaltfläche "Bearbeiten" angezeigt.

Kapitel 15

Dokumentieren von Mapping-Projekten

15 Dokumentieren von Mapping-Projekten

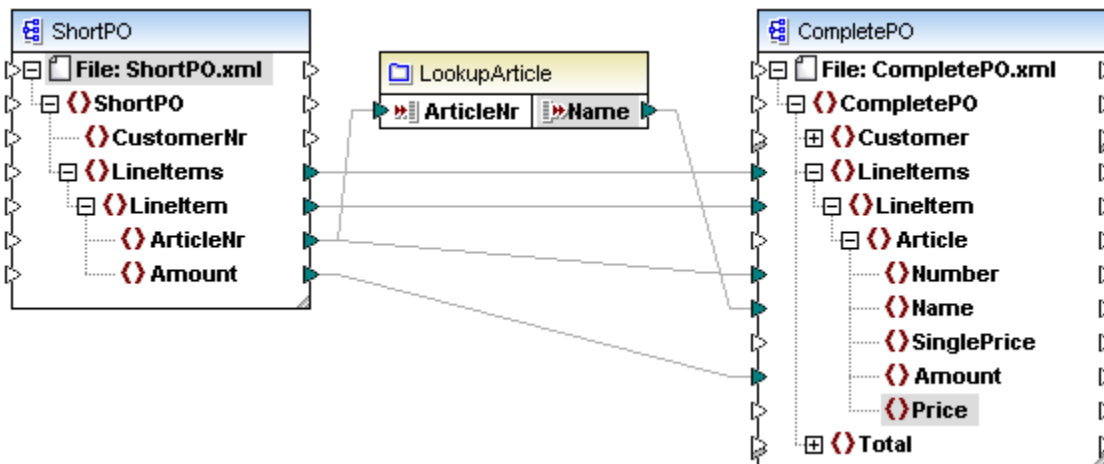
Mit dem Befehl **Dokumentation generieren** wird detaillierte Dokumentation zu Ihrem Mapping in HTML, MS Word, RTF oder PDF generiert. Die mit diesem Befehl generierte Dokumentation kann jederzeit verändert und verwendet werden; eine eigene Genehmigung von Altova ist nicht erforderlich.

Die Dokumentation wird für die Komponenten generiert, die Sie im Dialogfeld "Dokumentation generieren" auswählen. Zur Auswahl stehen entweder das vordefinierte festgelegte Design oder die Verwendung eines StyleVision SPS. Bei Verwendung eines StyleVision SPS können Sie das Design der generierten Dokumentation nach Ihren Wünschen gestalten. Eine Anleitung dazu finden Sie im Abschnitt [Benutzerdefiniertes Design](#).

Anmerkung: Um ein SPS für die Generierung der Mapping-Dokumentation verwenden zu können, müssen Sie StyleVision auf Ihrem Rechner installiert haben. Miteinander in Zusammenhang stehende Elemente werden normalerweise in der Bildschirmausgabe durch Hyperlinks miteinander verknüpft, so dass Sie von Komponente zu Komponente navigieren können.

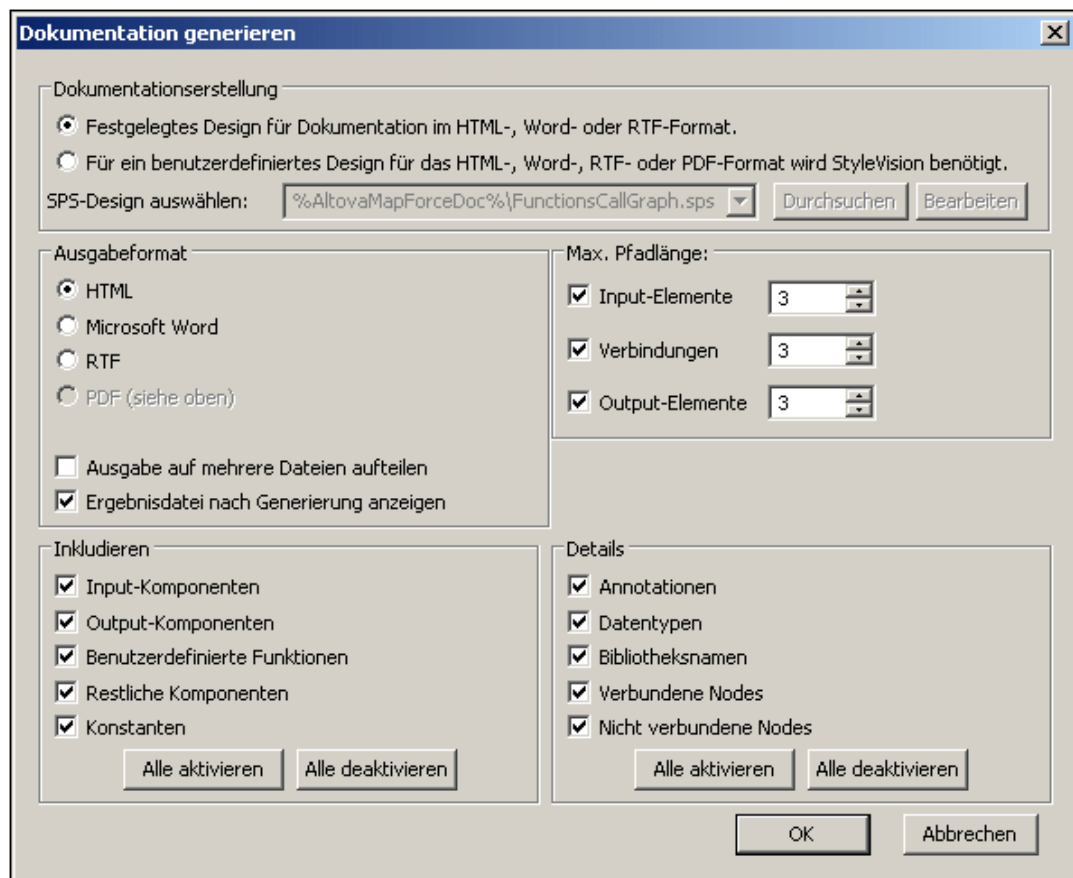
Um Dokumentation im MS Word-Format zu generieren, muss MS Word (Version 2000 oder höher) installiert sein.

In der Abbildung unten sehen Sie einen Ausschnitt aus der Datei **Lookup-standard.mfd** aus dem Ordner **...MapForceExamples**.



Nach Öffnen einer Mapping-Datei, z.B. `Lookup-standard.mfd`:

- Wählen Sie die Menüoption **Datei | Dokumentation generieren**. Daraufhin wird das Dialogfeld "Dokumentation generieren" geöffnet. In der Abbildung unten sehen Sie die Standardeinstellungen für das Dialogfeld.



Dokumentationserstellung

- Wählen Sie die Option "Festgelegtes Design für..", um die vordefinierte Dokumentationsvorlage zu verwenden.
- Wählen Sie die Option "Für ein benutzerdefiniertes Design...", um ein mit StyleVision erstelltes StyleVision Power Stylesheet zu verwenden. Die SPS-Dateien stehen im Ordner ...**My Documents\Altova\MapForce2012\Documentation\MapForce** zur Verfügung.
- Klicken Sie auf **Durchsuchen**, um zu einer vordefinierten SPS-Datei zu navigieren.
- Klicken Sie auf **Bearbeiten**, um StyleVision zu starten und das ausgewählte SPS in einem StyleVision-Fenster zu öffnen.

Im Ordner ...MapForce2012\Documentation\MapForce stehen die folgenden vordefinierten SPS Stylesheets zur Verfügung:

- [FunctionCallGraph.sps](#) - zeigt den Aufrufgraphen des Hauptmappings und aller benutzerdefinierten Funktionen.
- [FunctionsUsedBy.sps](#) - zeigt, welche Funktionen direkt oder indirekt im Mapping verwendet werden.
- [ImpactAnalysis.sps](#) - listet jeden einzelnen Quell- und Ziel-Node sowie die Route, die über die verschiedenen Funktionen zum Ziel-Node genommen wird, auf.
- [OverallDocumentation.sps](#) - zeigt alle Nodes, Verbindungen, Funktionen und Ziel-Nodes. Diese Option bietet die größte Detailliertheit und ist mit der vordefinierten Ausgabe "Festgelegtes Design ..." identisch.

Ausgabeformat

- Hier wird das Ausgabeformat definiert: entweder HTML, Microsoft Word, RTF oder PDF.

Bei Generierung der Dokumentation mit dem festgelegten Design werden Microsoft Word-Dokumente mit der Dateierweiterung **.doc** erzeugt. Bei Generierung unter Verwendung eines StyleVision SPS werden Dokumente mit der Dateierweiterung **.docx** erzeugt.

Das PDF-Ausgabeformat steht nur zur Verfügung, wenn Sie zum Generieren der Dokumentation ein StyleVision SPS verwenden.

- Wählen Sie die Option **Ausgabe auf mehrere Dateien aufteilen**, wenn Sie die Input-, Output-, Konstanten-Komponente und benutzerdefinierten Funktionen von der Dokumentation der Bibliothekskomponenten trennen möchten. In festgelegten Designs werden automatisch Links zwischen mehreren Dokumenten erstellt.
- Die Option **Ergebnisdatei nach Generierung anzeigen** ist für alle Ausgabeoptionen aktiviert. Wenn sie aktiv ist, werden die Ergebnisdateien im Standard-Browser (HTML-Ausgabe), in MS Word (MS Word-Ausgabe) und der Standardapplikation für RTF-Dateien (RTF-Ausgabe) angezeigt.

Maximale Pfadlänge

Über diese Option können Sie die maximale Pfadlänge definieren, die für Datenelemente angezeigt werden soll.

Beispiel: .../ShortPO/LinItems/LinItem. Dies wäre die maximale Länge bei der Standardeinstellung 3.

Inkludieren

- Hier können Sie definieren, welche Komponenten in der Dokumentation aufscheinen sollen.

Details

Dient zum Definieren der einzelnen Details, die in der Dokumentation aufscheinen sollen

- Wenn Sie "Bibliotheksnamen" aktivieren, wird das Präfix "core" in die oben gezeigten Funktionen eingefügt.
- Sie können nun sowohl verbundene als auch nicht verbundene Nodes dokumentieren

Anmerkung:

Mit den Schaltflächen **Alle aktivieren/deaktivieren** können Sie alle Kontrollkästchen dieser Gruppe aktivieren bzw. deaktivieren.

Wenn Sie die oben gezeigten Standardeinstellungen verwendet haben, werden Sie bei Klicken auf **OK** nach dem Namen für die Ausgabedatei und dem Speicherpfad gefragt. Unten sehen Sie einen Ausschnitt der mit dem festgelegten Design generierten Dokumentation. Beachten Sie, dass es sich hierbei um eine Datei mit einer einzigen Ausgabe handelt.

In dieser Tabelle sehen Sie die Verbindungen **von** der Quellkomponente zu der/den Zielkomponente(n).

Mapping **lookup-standard**

(C:\Documents and Settings\Wy\My Documents\Altova\MapForce2011\MapForceExamples\Lookup-standard.mfd)

Input **ShortPO** ([ShortPO.xsd](#))

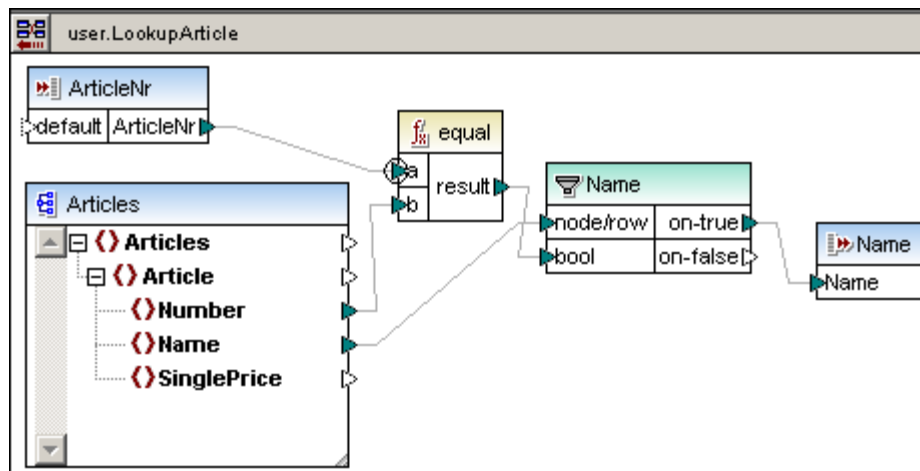
Nodes	Connections	
File: ShortPO.xml Type: string		
ShortPO Type: restriction of xs:anyType [0..1]		
ShortPO/CustomNr Type: xs:integer		
ShortPO/LinItems Type: restriction of xs:anyType	direct	CompletePO/LinItems Type: restriction of xs:anyType
ShortPO/LinItems/LinItem Type: restriction of xs:anyType [1..∞]	direct	CompletePO/LinItems/LinItem Type: restriction of xs:anyType [1..∞]
.../LinItems/LinItem/ArticleNr Type: xs:integer	direct	.../LinItem/Article/Number Type: xs:integer
	user.LookupArticle => ArticleNr Name =>	.../LinItem/Article/Name Type: xs:string
.../LinItems/LinItem/Amount Type: xs:integer	direct	.../LinItem/Article/Amount Type: xs:integer

Die Reihenfolge, in der die Komponenten dokumentiert sind, ist die folgende: Input, Output, Konstante, benutzerdefinierte Funktionen und schließlich Bibliotheksfunktionen.

Beispiel: **Input-Komponente** ShortPO:

- Die ersten beiden Datenelemente ShortPO und CustomNr sind mit keinem Datenelement in der Zielkomponente verbunden, daher ist die Spalte "Connections" leer.
- **ShortPO/LinItems** ist direkt mit **CompletePO/LinItems** in der Zielkomponente verbunden.
- **/LinItems/LinItem/ArticleNr** hat zwei Konnektoren:
 - direkt zu **LinItem/Article/Number** in der Zielkomponente
 - zur benutzerdefinierte Funktion **LookupArticle** mit ArticleNr als Input-Parameter und Name als dem Output-Parameter der benutzerdefinierten Funktion.

Unten sehen Sie den Inhalt der benutzerdefinierten Funktion.



Output-Komponente CompletePO: In dieser Tabelle sehen Sie die Verbindungen von der/den Quellkomponente(n) zur Zielkomponente.

Output **CompletePO** ([CompletePO.xsd](#))

Connections		Nodes
		File: CompletePO.xml Type: string
		CompletePO Type: restriction of xs:anyType [0..1]
		CompletePO/Customer Type: restriction of xs:anyType
ShortPO/LinItems Type: restriction of xs:anyType	direct	CompletePO/LinItems Type: restriction of xs:anyType
ShortPO/LinItems/LinItem Type: restriction of xs:anyType [1..∞]	direct	CompletePO/LinItems/LinItem Type: restriction of xs:anyType [1..∞]
		.../LinItems/LinItem/Article Type: extension of ArticleType
.../LinItems/LinItem/ArticleNr Type: xs:integer	direct	.../LinItem/Article/Number Type: xs:integer
.../LinItems/LinItem/ArticleNr Type: xs:integer	user.LookupArticle => ArticleNr Name =>	.../LinItem/Article/Name Type: xs:string

- Die ersten beiden Datenelemente CompletePO und CompletePO/Customer sind mit keinem Datenelement in der Quellkomponente verbunden, daher ist die Spalte "Connections" leer.
- CompletePO/LinItems** ist direkt mit **ShortPO/LinItems** in der Quellkomponente verbunden.
- LinItem/Article/Name ist mit der benutzerdefinierten Funktion **LookupArticle**, verbunden, wobei LinItems/LinItem/ArticleNr das Quelldatenelement ist.

User-defined function defines**user.LookupArticle**Input **Articles** ([Articles.xsd](#))

Nodes	Connections	
File: Articles.xml Type: string		
Articles Type: restriction of xs:anyType [0..1]		
Articles/Article Type: ArticleType [1..∞]		
Articles/Article/Number Type: xs:integer	core.equal => b result => core.filter => bool on-true =>	core.output => Name Type: string
Articles/Article/Name Type: xs:string	core.filter => node/row on-true =>	core.output => Name Type: string
Articles/Article/SinglePrice Type: xs:decimal		

Input **(required) core.ArticleNr**

Nodes	Connections	
ArticleNr Type: string	core.equal => a result => core.filter => bool on-true =>	core.output => Name Type: string

15.1 Mitgelieferte SPS Stylesheets

Funktionsaufrufgraphen - PersonListByBranchOffice.mfd

This report shows call graphs of the main mapping and all user-defined functions.

[Show all functions](#) [Collapse all functions](#)

Main mapping

```

|---core.equal
|---core.filter
|---user.LookupPerson
|   |---core.filter
|   |---user.EqualAnd
|   |   |---core.equal
|   |   |---core.logical-and
|   |---user.Person2Details
|   |---core.concat

```

user.LookupPerson

```

|---core.filter
|---user.EqualAnd
|   |---core.equal
|   |---core.logical-and
|---user.Person2Details
|   |---core.concat

```

user.EqualAnd

```

|---core.equal
|---core.logical-and

```

user.Person2Details

```

|---core.concat

```

Funktionen verwendet von - PersonListByBranchOffice.mfd

Library **core**

Function	Directly used by	Indirectly used by
core.equal	Main mapping user.EqualAnd	user.LookupPerson
core.filter	Main mapping user.LookupPerson	
core.logical-and	user.EqualAnd	Main mapping user.LookupPerson
core.concat	user.Person2Details	Main mapping user.LookupPerson

Library **user**

Function	Directly used by	Indirectly used by
user.LookupPerson	Main mapping	
user.EqualAnd	user.LookupPerson	Main mapping
user.Person2Details	user.LookupPerson	Main mapping

Analyse der Auswirkungen - PersonListByBranchOffice.mfd

This report lists every input and output node connection independently and is perfect for further impact analysis with modelling tools.

Input Node	Functions	Output Node
OfficeName	core.equal, core.filter	PersonList
OfficeName	user.LookupPerson	PersonList/Person/Details
BranchOffices/Office	core.filter	PersonList
BranchOffices/Office/Name	core.equal, core.filter	PersonList
BranchOffices/Office/Contact		PersonList/Person
.../Office/Contact/first		PersonList/Person/First
.../Office/Contact/first	user.LookupPerson	PersonList/Person/Details
.../Office/Contact/last		PersonList/Person/Last
.../Office/Contact/last	user.LookupPerson	PersonList/Person/Details

Overall Documentation - PersonListByBranchOffice.mfd

Mapping **PersonListByBranchOffice.mfd**Input **core.OfficeName**

Nodes	Connections	
OfficeName Type: string	<u>core.equal => a result =></u> <u>core.filter => bool on-true =></u>	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
	<u>user.LookupPerson =></u> <u>Office Name result =></u>	PersonList/Person/Details Type: xs:string [0..1]

Input **BranchOffices** (BranchOffices.xsd)

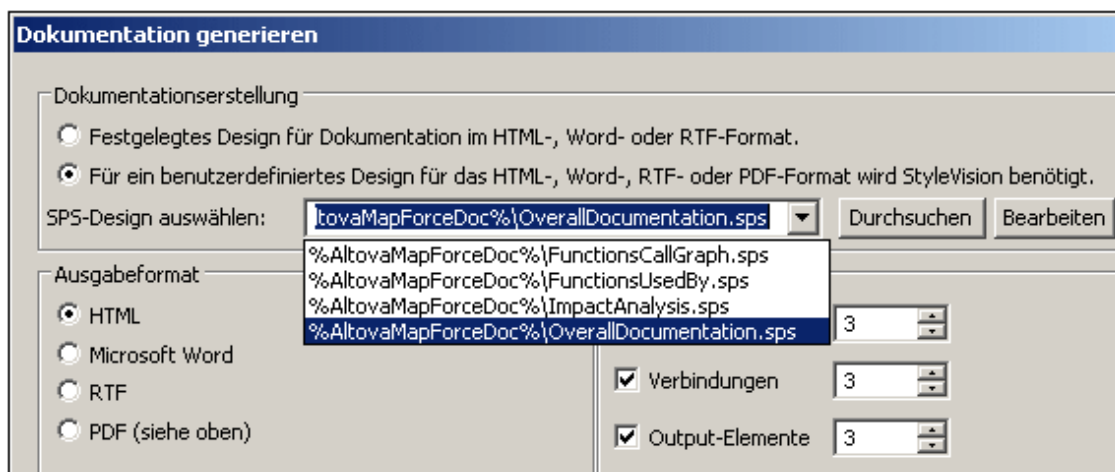
Nodes	Connections	
File: BranchOffices.xml Type: string		
BranchOffices Type: restriction of xs:anyType [0..1]		
BranchOffices/Name Type: restriction of xs:string		
BranchOffices/Office Type: restriction of xs:anyType [0..∞]	<u>core.filter => node/row on-true =></u>	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
BranchOffices/Office/Name Type: restriction of xs:string	<u>core.equal => b result =></u> <u>core.filter => bool on-true =></u>	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons

15.2 Benutzerdefiniertes Design

Anstelle eines festgelegten Designs können Sie auch ein benutzerdefiniertes Design für die MapForce Dokumentation erstellen. Das benutzerdefinierte Design wird in einem StyleVision SPS erstellt. Beachten Sie, dass 4 vordefinierte SPS Stylesheets mit MapForce mitgeliefert werden. Nähere Informationen dazu finden Sie unter [Dokumentieren von Mapping-Projekten](#).

Definieren des SPS zur Verwendung der MapForce-Dokumentation

Welches SPS zur Generierung der Dokumentation verwendet werden soll, wird im Dialogfeld "Dokumentation generieren" (Aufruf über **Datei | Dokumentation generieren**) definiert. Aktivieren Sie das Optionsfeld "Für ein benutzerdefiniertes Design..." und klicken Sie anschließend auf den Dropdown-Pfeil der Auswahlliste um die gewünschte Datei auszuwählen. Standardmäßig ist der Eintrag **OverallDocumentation.sps** ausgewählt.



Diese vordefinierten SPS-Dateien befinden sich im Ordner ...MapForce2012\Documentation\MapForce.

Bitte beachten Sie:

Um ein SPS zur Generierung der Dokumentation verwenden zu können, muss StyleVision auf Ihrem Rechner installiert sein.

Erstellen des SPS

Ein StyleVision Power Stylesheet (oder SPS) wird mit Hilfe des Altova-Produkts [StyleVision](#) erstellt. Ein SPS zur Generierung von MapForce Dokumentation muss auf dem XML-Schema basieren, in dem die Struktur des XML-Dokuments, das die MapForce-Dokumentation enthält, definiert ist.

Dieses Schema hat den Namen **MapForceDocumentation.xsd** und ist im MapForce-Installationspaket enthalten. Es ist im Ordner ...My Documents\Altova\MapForce2012\Documentation gespeichert.

Bei der Erstellung des SPS-Designs in StyleVision werden Nodes aus dem Schema MapForceDocumentation.xsd in das Design platziert und diesen Nodes werden Stile und Eigenschaften zugewiesen. Beachten Sie, dass die Datei MapForceDocumentation.xsd die Datei Documentation.xsd aus dem übergeordneten Ordner inkludiert.

Des Weiteren können zusätzliche Komponenten wie z.B. Links und Bilder ebenfalls zum SPS-Design hinzugefügt werden. Eine Anleitung zur Erstellung eines SPS Designs in StyleVision

finden Sie im Benutzerhandbuch zu StyleVision.

Der Vorteil bei der Generierung der Mapping-Dokumentation anhand eines SPS ist, dass Sie völlig freie Hand bei der Gestaltung der Dokumentation haben. Beachten Sie außerdem, dass die PDF-Ausgabe der Dokumentation nur bei Verwendung eines SPS zur Verfügung steht; bei Verwendung des festgelegten Designs steht die PDF-Ausgabe nicht zur Verfügung.

Kapitel 16

Nullwerte / Nillable Werte

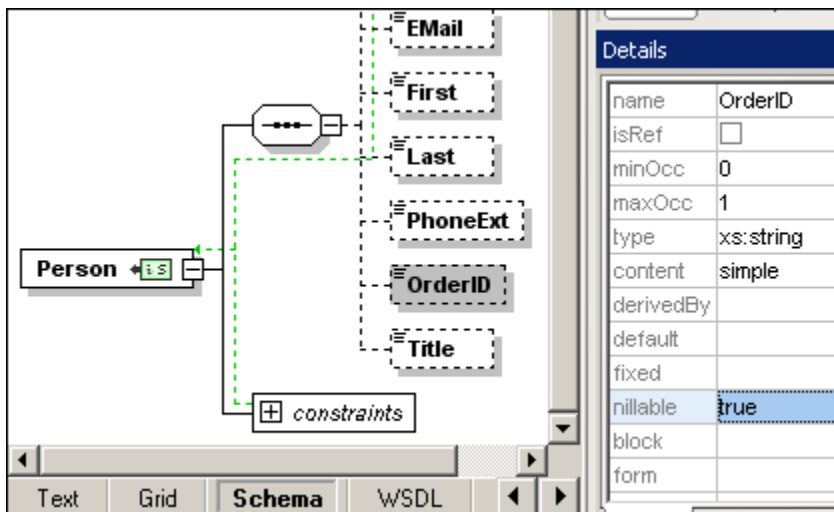
16 Nullwerte / Nillable Werte

Nach der XML-Schema-Spezifikation dürfen gültige Elemente auch leer sein, d.h. keinen Inhalt enthalten, (auch wenn ein Content Type leeren Inhalt nicht zulässt), wenn für das entsprechende Element im Schema das Attribut `nillable="true"` definiert ist.

Das bedeutet, dass im XML-**Instanzdokument** mit Hilfe eines **Attributs** `xsi:nil="true"` angegeben wird, dass das Element zwar vorhanden ist, dass der **Inhalt** dieses Elements aber Null ist.

Beachten Sie, dass dies nur für **Elementwerte**, nicht aber für Attributwerte gilt. Das Element mit `xsi:nil="true"` hat zwar vielleicht keinen Inhalt oder enthält keinen Text, kann aber weitere Attribute haben.

Das Attribut **nillable="true"** ist im XML-**Schema** definiert und kann sowohl in der Quell- als auch in der Zielkomponente vorkommen.



Das Attribut `xsi:nil="true"` ist in der XML-**Instanzdatei** definiert und kommt nur in der Instanzdatei vor.

```

14  <Person>
15    <PrimaryKey>2</PrimaryKey>
16    <ForeignKey>1</ForeignKey>
17    <Email>biff@amail.com</Email>
18    <First>biff</First>
19    <Last>bander</Last>
20    <PhoneExt>22</PhoneExt>
21    <OrderID xsi:nil="true"/>
22    <Title>IT services</Title>
23  </Person>

```

Das Attribut `xsi:nil` wird im grafischen MapForce-Mapping nicht explizit angezeigt, da es in den meisten Fällen automatisch verarbeitet wird. Ein auf Null gesetzter ("nilled") Node, also ein Node, für den das Attribut `xsi:nil="true"` gesetzt wurde, ist zwar vorhanden, hat aber keinen Inhalt.

Nillable Elemente als Quellkomponente

Immer, wenn in einem Mapping Daten **aus** einem XML- oder XBRL-Element ausgelesen werden, das auf Null gesetzt wurde, wird das Attribut `xsi:nil` automatisch überprüft. Wenn der Wert von `xsi:nil` "true" ist, so wird der Inhalt als nicht vorhanden behandelt.

Bei Erstellung eines **zielorientierten** Mappings von einem nillable Quellelement auf ein nillable Zielelement mit **simpleType** Inhalt (ein einziger Wert mit optionalen Attributen aber ohne Child-Elemente), bei dem `xsi:nil` für ein Quellelement definiert ist, wird das Attribut `xsi:nil` in das Zielelement eingefügt z.B. `<OrderID xsi:nil="true"/>`.

Bei Erstellung eines **Alles kopieren**-Mappings von einem nillable Quellelement auf ein nillable Zielelement, in dem `xsi:nil` für ein Quellelement definiert ist, wird das Attribut `xsi:nil` in das Zielelement z.B. eingefügt `<OrderID xsi:nil="true"/>`.

Verwenden von Funktionen

Wenn Sie die "exists"-Funktion mit einem auf Null gesetzten Quellelement verbinden, wird für alle Elemente "true" zurückgegeben, da der **Element**-Node ja tatsächlich existiert, auch wenn er keinen Inhalt enthält.

Um explizit zu überprüfen, ob das Attribut `xsi:nil` bei einem Quellelement auf "true" gesetzt ist, verwenden Sie die Funktion **is-xsi-nil**. Es gibt für auf Null gesetzte Elemente "true" zurück und für andere Nodes "false".

Die Verwendung von Funktionen, die einfache Werte erwarten (z.B. multiply, concat), für Elemente, für die `xsi:nil` definiert wurde, hat kein Ergebnis zur Folge, da kein Elementinhalt vorhanden ist und kein Wert extrahiert werden kann. Diese Funktionen verhalten sich, als wäre der Quell-Node nicht vorhanden.

Um einen auf Null gesetzten (nicht existierenden) Quellelementwert durch einen anderen Wert zu ersetzen, verwenden Sie die Funktion **substitute-missing**.

Mappen von `xsi:nil` auf ein Datenbankfeld

Wenn Sie ein auf Null gesetztes XML-Element auf eine Datenbankspalte mappen, wird ein Wert NULL geschrieben. Wenn Sie ein Datenbankfeld in jedem Fall auf Null setzen möchten, können Sie dies auch mit Hilfe der Funktion **set-null** tun.

Auf Null gesetzte Elemente als Zielkomponente

Bei Erstellung eines **zielorientierten** Mappings von einem nillable Quellelement auf ein nillable Zielelement mit **simpleType** Inhalt (ein einziger Wert mit optionalen Attributen aber ohne Child-Elemente), bei dem `xsi:nil` für ein Quellelement definiert ist, wird das Attribut `xsi:nil` in das Zielelement eingefügt z.B. `<OrderID xsi:nil="true"/>`.

Wenn das Attribut `xsi:nil="true"` im XML-Quellelement **nicht** definiert wurde, so wird der Elementinhalt auf die übliche Weise auf das Zielelement gemappt.

Beim Mappen eines nillable Zielelements vom Typ **complex type** (mit Child-Elementen) wird das Attribut `xsi:nil` **nicht** automatisch geschrieben, da MapForce zum Zeitpunkt, zu dem es die Attribute des Elements schreibt, nicht weiß, ob Child-Elemente folgen. Definieren Sie eine "**Alles kopieren**"-Verbindung, um das Attribut `xsi:nil` aus dem Quellelement zu kopieren.

Beim Mappen einer **leeren Sequenz** oder eines Datenbank-NULL-Werts auf ein Zielelement wird das Element gar nicht erstellt - unabhängig davon, ob es nillable ist oder nicht.

Verwendung von Funktionen

Um die Erstellung eines leeren Zieldatenelements mit `xsi:nil` auf "true" gesetzt zu erzwingen,

verbinden Sie die Funktion **set-xsi-nil** direkt mit dem Zieldatenelement. Dies funktioniert bei Zieldatenelementen vom Typ "simpleType" und "complexType".

Verwenden Sie die Funktion **substitute-missing-with-xsi-nil** um `xsi:nil` in die Zielkomponente einzufügen, wenn in Ihrer Mapping-Quellkomponente kein Wert zur Verfügung steht. Dies kann vorkommen, wenn der Quell-Node nicht vorhanden ist oder wenn bei einer Berechnung (z.B. einer Multiplikation) ein auf Null gesetzter Quell-Node beteiligt war, aufgrund dessen kein Ergebnis erzeugt wurde.

Diese Funktion kann auch dazu verwendet werden, um anhand von Datenbanknullwerten mit Hilfe von `xsi:nil="true"` Elemente zu erstellen. Diese Funktion funktioniert nur bei Nodes mit Inhalt vom Typ `simpleType`.

Um `xsi:nil` bei einem Element vom Typ **complexType** zu erstellen, gehen Sie folgendermaßen vor:

1. Erstellen Sie das Mapping wie gewohnt für den Fall, in dem der Ziel-Node Inhalt enthält.
2. Fügen Sie eine Filterkomponente in die Verbindung zum Ziel-Node ein und definieren Sie die Bedingung, unter der Inhalt vorhanden ist.
3. Verwenden Sie die Funktion "Duplikat erzeugen" um Ihren Ziel-Node zu kopieren.
4. Verbinden Sie die Funktion "set-xsi-nil" über eine weitere Filterkomponente mit dem kopierten Ziel-Node.
5. Fügen Sie eine "logical-not"-Funktion ein, um die Bedingung zu negieren und verbinden Sie sie mit dem Filter zwischen "set-xsi-nil" und dem kopierten Ziel-Node.

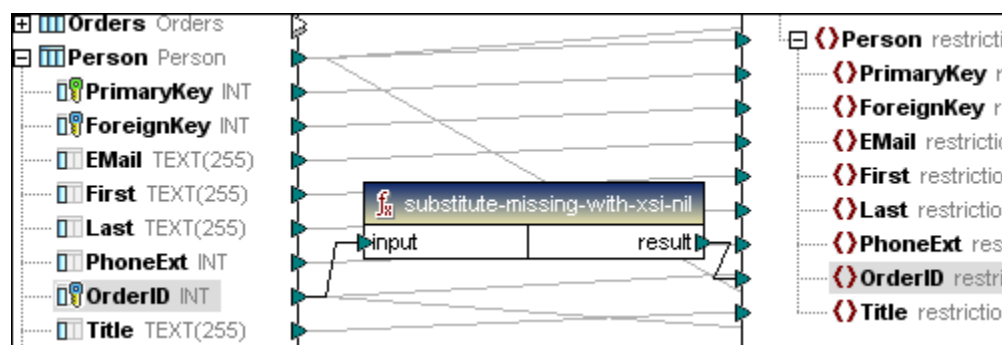
Mappen von Null-Feldern von Datenbankkomponenten

Wenn ein Null-Datenbankfeld auf ein nillable-Element eines XML-Schemas gemappt wird, dann werden nur diejenigen Zielemente generiert, die auch tatsächlich Datenbankdaten enthalten. Elemente von *NULL*-Datenbankfeldern werden in der Zielkomponente gar nicht angelegt.

Wenn Sie die Node-Funktion "exists" mit einem solchen Quellelement verbinden, erhalten Sie den Wert "false" für die **Null-Felder**.

So erzwingen Sie die Erstellung aller Elemente in der Zielkomponente:

Verwenden Sie die Funktion **substitute-missing-with-xsi-nil** aus der Core-Bibliothek (node functions).



Es werden nun für alle Datenbankfelder Zielemente erstellt.

- Alle (fehlenden)/Null-Datenbankfelder enthalten nun im Zielement `<OrderID xsi:nil="true"/>`.
- Vorhandene Daten aus Datenbankfeldern werden direkt auf das Zielement gemappt z.B. `<OrderID>1</OrderID>`.

Um die Null-Felder einer Datenbankkomponente zu finden/anzuzeigen, klicken Sie auf die Schaltfläche "Datenbankabfrage" und führen Sie an der/den Datenbanktabelle(n) eine Abfrage

durch. Null-Felder werden im Ergebnisfenster als *[NULL]* angezeigt.

 						
	E-Mail	First	Last	PhoneExt	OrderID	Title
1	v.callaby@nanonull.com	Vernon	Callaby	582	[NULL]	Office Manager
2	f.further@nanonull.com	Frank	Further	471	[NULL]	Accounts Recei
3	l.matise@nanonull.com	Loby	Matise	963	1	Accounting Man
4	j.firstbread@nanonull.com	Joe	Firstbread	621	[NULL]	Marketing Manag

Kapitel 17

Datenbanken und MapForce

17 Datenbanken und MapForce

Altova Website:  [Datenbank-Mapping](http://www.altova.com/Datenbank-Mapping)

MapForce bietet umfangreiche Unterstützung für Datenbank-Mappings, u.a. auch Mappen von Datenbankdaten und XML-Dokumenten Flat Files, EDI, Excel 2007+, XBRL, Web Services, und sogar anderen Datenbankformaten.

MapForce berücksichtigt Einschränkungen durch Primär- und Sekundärschlüssel und erzeugt auch Transaktionen, um die Datenintegrität sicherzustellen.

Bitte beachten Sie:

Für einen Zugriff auf eine Datenbank müssen Sie eine der folgenden Optionen verwenden: Java, C#, C++ oder BUILTIN (den Built-In-Ausführungsprozessor). **XQuery** -Code kann nur für XML-Datenquellen und -ziele generiert werden.

Derzeit werden die folgenden Datenbanken (und Verbindungsarten) unterstützt:

- Microsoft Access; Version 2003 / 2007
- Microsoft SQL Server Versionen 2000, 2005 und 2008
- Oracle Versionen 9i, 10g und 11g
- MySQL 4.x und 5.x
- PostgreSQL 8.0, 8.1, 8.2, 8.3
- Sybase 12
- IBM DB2 Version 8.x und 9
- IBM DB2 für i 5.4
- IBM DB2 für i 6.1

Bitte beachten Sie:

MapForce 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.

Stellen Sie bei Installation der **64-Bit** Version von **MapForce** sicher, dass Sie Zugriff auf die 64-Bit-Datenbanktreiber haben, die für die jeweilige Datenbank, zu der Sie eine Verbindung herstellen möchten, benötigt werden.

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.

17.1 Installation von Datenbank-Clients / -Treibern

Dieses Dokument soll dem Endbenutzer bei der Installation verschiedener Datenbank-Clients behilflich sein und ist nicht als genaue Anleitung zu verstehen sondern als Richtlinie. Für die Richtigkeit der Angaben wird keine Gewährleistung übernommen.

Wir empfehlen Ihnen, Client Software von Ihrem Datenbank Administrator installieren zu lassen und als Endbenutzer keine manuelle Installation vorzunehmen.

In den folgenden Abschnitten finden Sie Anleitungen für die Installation der einzelnen Datenbank Clients / Treiber.

Voraussetzung:

Die jeweilige Datenbank ist bereits installiert und Sie haben LAN-Zugriff auf den Datenbank-Server.

17.1.1 SQL Server

Microsoft SQL Server Datenbanken:

[SQL Server 2000](#)

[SQL Server 2005](#)

[SQL Server 2008](#)

SQL Server 2000

- Bitten Sie Ihren Datenbankadministrator, Ihnen den Microsoft SQL Server DB Client zur Verfügung zu stellen und installieren Sie ihn.
- Starten Sie den Enterprise Manager, rechtsklicken Sie auf "SQL SERVER Group", wählen Sie "New SQL Server Registration" und fügen Sie einen Server hinzu.
- Bitten Sie Ihren Datenbankadministrator um die Login-Daten, also Benutzernamen und Passwort, um eine Verbindung herzustellen.
- Wählen Sie "SQL Server authentication", geben Sie den Benutzernamen und das Passwort ein und fügen Sie anschließend den SQL Server zu einer bestehenden SQL Server Gruppe hinzu.
- Erweitern Sie die linke Baumstruktur und wählen Sie eine Datenbank: Microsoft SQL Server->SQL Server Group->{Server Name}->Databases->{Datenbankname}

SQL Server 2005

- Bitten Sie Ihren Datenbankadministrator, Ihnen den Microsoft SQL Server DB Client zur Verfügung zu stellen und installieren Sie ihn.
- Wählen Sie im Dialogfeld "Components to install" die gewünschten Client- und Dokumentkomponenten aus
- Wählen Sie keine Komponente aus und klicken Sie auf "Advanced"
- Wählen Sie "Client Components" (der gesamte Inhalt wird installiert)
- Wählen Sie "Documentation, Samples, and..." (der gesamte Inhalt wird installiert)
- Klicken Sie auf "Next" und "Install"

SQL Server 2008

- Bitten Sie Ihren Datenbankadministrator, Ihnen den Microsoft SQL Server DB Client zur Verfügung zu stellen und installieren Sie ihn.

17.1.2 Oracle

Oracle-Datenbanken:

[Oracle 9](#)
[Oracle 10](#)
[Oracle 11](#)

Oracle 9i

- Bitten Sie Ihren Datenbankadministrator, Ihnen das Oracle Client Administrator-Paket zur Verfügung zu stellen und installieren Sie es.
- Kopieren Sie die beiden Dateien **sqlnet.ora** und **tnsnames.ora** aus dem Paket in ... \ORACLE\ora92\network\ADMIN.
- Bitten Sie Ihren Datenbankadministrator um die Login-Daten, also Benutzernamen und Passwort, um eine Verbindung zur Datenbank herzustellen.

Downloads finden Sie unter <http://www.oracle.com/technology/software/products/oracle9i/index.html>.

Oracle 10g

- Bitten Sie Ihren Datenbankadministrator, Ihnen das Oracle Client Administrator-Paket zur Verfügung zu stellen und installieren Sie es.
- Während der Installation wird der Oracle Net Konfigurationsassistent gestartet.
- Klicken Sie in jedem der Dialogfelder auf "Next" ohne die Standardeinstellungen zu ändern.
- In einem Dialogfeld werden Sie aufgefordert den **Service-Namen**, anzugeben, in einem anderen den **Server-Namen**. Geben Sie die beiden Informationen ein.
- Verwenden Sie Standard Port. Nr. 1521 oder fragen Sie Ihren Administrator nach der richtigen Port-Nummer.
- Bitten Sie Ihren Datenbankadministrator um die Login-Daten, also Benutzernamen und Passwort, um eine Verbindung zur Datenbank herzustellen.

Downloads finden Sie unter <http://www.oracle.com/technology/software/products/database/index.html>.

Oracle 11g

- Bitten Sie Ihren Datenbankadministrator, Ihnen das Oracle Client Administrator-Paket zur Verfügung zu stellen und installieren Sie es.
- Befolgen Sie die Installationsanweisungen.
- Bitten Sie Ihren Datenbankadministrator um die Login-Daten, also Benutzernamen und Passwort, um eine Verbindung zur Datenbank herzustellen.

Downloads finden Sie unter <http://www.oracle.com/technology/software/products/database/index.html>.

Installation eines Oracle Client

Im Folgenden wird beschrieben, wie Sie eine neue Verbindung zu einer Oracle Datenbank in Ihrem lokalen Netzwerk herstellen. Der Assistent für die Namenskonfiguration des lokalen Netzwerks geht bei der Erstinstallation des Oracle Client bei der Installation des Netzwerkdiensts auf dieselbe Weise vor.

1. Wählen Sie die Menüoption **Programme | Oracle - OraHome92 | Configuration and**

migration tools | Net Configuration Assistant.

Daraufhin wird der Oracle Net-Konfigurationsassistent geöffnet.

2. Aktivieren Sie das Optionsfeld **Local Net Service Name Configuration** und klicken Sie auf "Next".
3. Klicken Sie auf **Add** um einen neuen Netzwerkdienstnamen hinzuzufügen und klicken Sie auf "Next".
4. Wählen Sie die installierte Oracle-Version aus, z.B. **Oracle 8i** oder höher und klicken Sie auf "Next".
5. Geben Sie den **"Service Name"** der gewünschten Datenbank ein, z.B. TestDB und klicken Sie auf "Next". Beim Service Namen der Datenbank handelt es sich im Allgemeinen um ihren globalen Datenbanknamen.
6. Wählen Sie das Netzwerkprotokoll (**network protocol**) zum Aufrufen der Datenbank aus, z.B. TCP und klicken Sie auf "Next".
7. Geben Sie den Host-Namen (**Host name**) des Rechners an, auf dem die Datenbank installiert ist, und geben Sie falls nötig, die Port-Nummer ein. Klicken Sie zum Fortfahren auf "Next".
8. Aktivieren Sie das Optionsfeld **Yes** um die Datenbankverbindung zu testen und klicken Sie auf "Next".
9. Sie können den Login-Parameter ändern, falls der Test nicht erfolgreich war. Klicken sie dazu auf die Schaltfläche "Change Login" und geben Sie den neuen Parameter ein. Klicken Sie zu Fortfahren auf "Next".
10. Geben Sie in das Feld **Net Service Name** den Namen des Netzwerkdiensts ein. Sie können jeden beliebigen Namen wählen. Dies ist der Name, den Sie in MapForce im Dialogfeld **Oracle login** in das Feld "Datenbank" eingeben werden. Klicken Sie zu Fortfahren auf "Next".
11. Damit ist die Konfiguration des Netzwerkdienstnamens abgeschlossen. Klicken Sie auf "Next", um das Dialogfeld zu schließen.

17.1.3 IBM DB2

IBM DB2-Datenbanken:

[DB2 Version 8 / 9](#)

[DB2 für i 5.4](#)

DB2 Version 8 / 9

Voraussetzung: JRE (Java Runtime Environment) 1.4.2 oder höher muss installiert sein.

- Bitten Sie Ihren Datenbankadministrator, Ihnen das IBM DB2 Client Installationspaket zur Verfügung zu stellen und installieren Sie es. Befolgen Sie die Installationsanweisungen.
- Starten Sie nach Abschluss der Installation den "Konfigurationsassistenten", um Ihre Datenbank zu konfigurieren.
Starten Sie IBMDB2->setup tools->Configuration Assistant.
Es erscheint die Meldung: "... Would you like to add a database now?".
Klicken Sie auf "Yes", daraufhin wird der "Add Database Wizard" gestartet.
Wählen Sie "Search the network", "Next", "Add system", "Discover" und wählen Sie den Systemnamen aus.
- Fügen Sie eine Datenbank hinzu:
Starten Sie IBMDB2->General administration Tools->Control Center.
Erweitern Sie die Baumstruktur: Control center->All Cataloged Systems->{Servername}->Instances->{instance_name}->Databases.
Rechtsklicken Sie auf Databases und wählen Sie den Menübefehl "Add".
Klicken Sie auf "Discover", wählen Sie eine Datenbank aus und klicken Sie auf "OK".
- Registrieren Sie die Datenbank als eine Datenquelle:
Starten Sie den Konfigurationsassistenten und wählen Sie eine Datenbank aus.
Wählen Sie in der linken Liste "data Source".
Wählen Sie "Register this database for ODBC" und aktivieren Sie das Kontrollkästchen "As user data source". Klicken Sie auf "Finish".
- Testen Sie die Verbindung zur Datenbank:
Starten Sie den Konfigurationsassistenten und wählen Sie eine Datenbank aus..
Wählen Sie den Menübefehl "Selected->Test connection".
Wählen Sie als Verbindungsart (connection type) CLI, ODBC aus.
Bitten Sie Ihren Datenbankadministrator um Benutzernamen und Passwort und klicken Sie auf "Test connection"
- Stellen Sie bitte sicher, dass Sie das Kontrollkästchen "Remember Password" aktiviert haben, da sonst die Einstellungen für die Verbindung zur Datenbank verloren gehen.

DB2 für i 5.4

Bitten Sie Ihren Datenbankadministrator um eine Anleitung, wie man den Datenbank-Server aufruft. Beachten Sie, dass iSeries der neue Name für die AS400 Computerserie ist.

17.1.4 MySQL

MySQL Datenbanken

[MySQL 4 / 5](#)

MySQL 4 / 5

Im Vergleich zu anderen Datenbanken bietet MySQL eine separate Treiberinstallation. Sie müssen keinen Client installieren.

Installation des ODBC-Treibers

- Installation des neuesten ODBC-Treibers (zum Zeitpunkt der Verfassung des Dokuments Version 5.1.xx)
- Starten Sie die Systemsteuerung und wählen Sie "Administrative tools"->"Data Source (ODBC)"
- Wählen Sie "User DSN" und klicken Sie auf "Add"
- Wählen Sie den neuesten "MySQL ODBC Driver" (zum Zeitpunkt der Verfassung des Dokuments Version 5.1.xx) und klicken Sie auf "Finish"
- Füllen Sie die Felder "Data Source Name", "Server", "User", "Password" und "Database" aus
- Klicken Sie auf "Test" und falls der Test erfolgreich verläuft, auf "OK"

Downloads finden Sie unter <http://dev.mysql.com/downloads/connector/odbc/>

17.1.5 PostgreSQL

PostgreSQL Datenbanken

[PostgreSQL 8.x Datenbanken](#)

PostgreSQL 8.x

Im Vergleich zu anderen Datenbanken bietet PostgreSQL eine separate Treiberinstallation. Sie müssen keinen Client installieren

- Installieren Sie den ODBC Treiber: Führen Sie **psqlodbc.msi** aus und befolgen Sie die Anweisungen

Downloads finden Sie unter <http://www.postgresql.org/ftp/odbc/versions/msi>

17.1.6 Sybase

Sybase Datenbanken:

[Sybase 12](#)

Sybase 12

Bitten Sie Ihren Datenbankadministrator, Ihnen das Sybase Client-Paket zur Verfügung zu stellen und installieren Sie es.

- Installieren Sie den Sybase Client und befolgen Sie die Installationsanweisungen
- Eventuell müssen Sie die Datei c:\sybase\ini\sql.ini bearbeiten
- Starten Sie die Systemsteuerung und wählen Sie "Administrative tools"->"Data Source (ODBC)"
 - Wählen Sie "User DSN" und klicken Sie auf "Add"
 - Wählen Sie den Treiber "Sybase ASE ODBC Driver" und klicken Sie auf "Finish"
 - Geben Sie den Namen der DSN-Datei in das Feld "Data Source Name" ein
 - Fügen Sie "Winsock" zu "Network Library Name" hinzu
 - Fügen Sie "<server name>,2048" zu "Network Address" hinzu
 - Fügen Sie "<database name>" zum Feld "Database Name" hinzu
 - Klicken Sie auf "Test". Ist der Test erfolgreich, klicken Sie auf "OK"

Nähere Informationen finden Sie unter

<http://www.sybase.com/products/databasemanagement/adaptiveserverenterprise>.

Die sql.ini Dateieinträge haben im Allgemeinen die folgende Form:

```
[<servername>]
master=TCP, <servername>,<portnumber>
query=TCP, <servername>,< portnumber>
```


17.1.7 Microsoft Access

Microsoft Access 2004 und 2007

Zur Verwendung dieser Datenbanken müssen keine Clients oder Treiber installiert werden. Sie benötigen nur direkten Zugriff über ein LAN.

17.1.8 Firebird

Firebird Datenbanken:

[Firebird 2.0.5 / 2.1.2](#)

Firebird 2.0.5 / 2.1.2

Für den Firebird ODBC Treiber muss eine Client DLL installiert werden. Installieren Sie daher den Firebird Client und anschließend den Treiber.

- Starten Sie das Windows Exe Installationsprogramm für Full Classic.
Sie können nur die Client Installation auswählen.
- Führen Sie die Windows Datei **Full Install.exe** aus, um den Treiber zu installieren.
- Starten Sie die Systemsteuerung und wählen Sie "Administrative tools"->"Data Source (ODBC)"
- Wählen Sie "User DNS" und klicken Sie auf "Add"
- Wählen Sie "Firebird Driver" und klicken Sie auf "Finish"
- Füllen Sie die Felder "Data Source Name", "Database", "Database Account" und "Password" aus
- Wählen Sie **fbClient.dll** aus der Client Installation in "Client" aus
- Klicken Sie auf "OK"

Client Download für Version 2.0.5

http://www.firebirdsql.org/index.php?op=files&id=engine_205

Client Download für Version 2.1.2

http://www.firebirdsql.org/index.php?op=files&id=engine_212

ODBC Treiber Download für beide Versionen

<http://www.firebirdsql.org/index.php?op=files&id=odbc>

17.2 Tutorial: Mappen von XML-Daten auf Datenbanken

In diesem Tutorial lernen Sie Folgendes:

- [Einrichten eines Mappings von XML auf eine Datenbank](#)
- [Einfügen von Daten in Datenbanktabellen](#)
- Aktualisieren
- Löschen
- Ignorieren
- Generieren von Datenbank Ausgabewerten

Beispieldateien für das Tutorial

In diesem Abschnitt verwendete Dateien:

Altova_Hierarchical.xsd	die Datei mit dem hierarchischen Schema mit Identity Constraints
Altova-cmpy.xml	die Datei mit den Altova-XML-Firmendaten
Altova.mdb	die Altova MS-Access-Datenbankdatei, die als Zieldatenbank verwendet wird

Alle diese Beispieldateien liegen im Ordner [...MapForceExamples](#). Bitte beachten Sie: In diesem Abschnitt werden wir viel mit der Datenbank **Altova.mdb** arbeiten, um die Funktionen von MapForce im Zusammenhang mit der Verwendung von Datenbanken als Ziel zu erläutern. Legen Sie bitte eine Sicherungskopie der Datei an, bevor Sie eines der hier beschriebenen Beispiele ausprobieren. Um dieselben Ergebnisse wie in den Tutorial-Beispielen zu erhalten, sollten Sie jeden Abschnitt von Grund auf neu beginnen, ohne, dass Sie an der Datenbank vorher Änderungen vorgenommen haben.

MapForce kann Daten auch von passwortgeschützten Access-Dateien mappen, wenn Sie über die Option **Beliebige ODBC** aus dem Dialogfeld "Quelldatenbank auswählen" hinzugefügt werden. Die Benutzer- und Passwordeinstellungen können im Assistenten eingegeben werden.

17.2.1 Einrichten von XML-auf-Datenbank Mappings

Das Einrichten eines Mappings von einer XML-Datei auf eine Datenbank erfolgt auf die gleiche Weise, wie das zuvor beschriebene Mapping. **Bitte beachten Sie:** Mappings zwischen Datenbankkomponenten können nicht erstellt werden, wenn Sie XSLT, XSLT2 oder XQuery als Zielsprache auswählen. XSLT unterstützt keine Datenbankabfragen.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man eine Datenbankkomponente hinzufügt und eine Verbindung mit der dazugehörigen Datenbank herstellt.

In diesem Abschnitt verwendete Befehle



Built-in Ausführungsprozessor: Dieser Befehl befindet sich in der Symbolleiste "Auswahl der Sprache" und im Menü **Ausgabe**. Klicken Sie auf den Befehl, um den Built-in Ausführungsprozessor als Ausgabeformat auszuwählen.



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



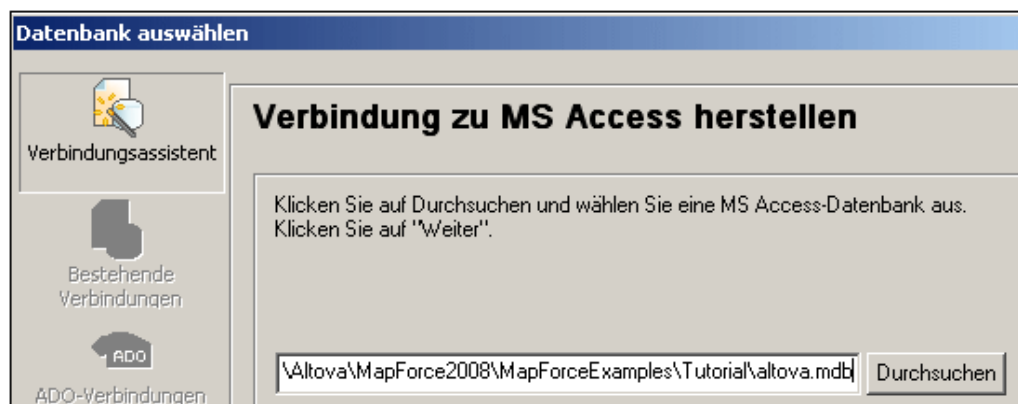
Datenbank einfügen: Dieser Befehl befindet sich in der Symbolleiste "Komponente einfügen" und im Menü **Einfügen**. Klicken Sie auf diesen Befehl, um ein Dialogfeld **Datenbank auswählen** zu öffnen, in dem Sie eine Datenbankverbindung definieren und auswählen können, welche Datenbanktabellen in das MapForce Mapping inkludiert werden sollen.

So richten Sie die Mapping-Umgebung ein und stellen eine Verbindung zu einer MS Access-Datenbank her:

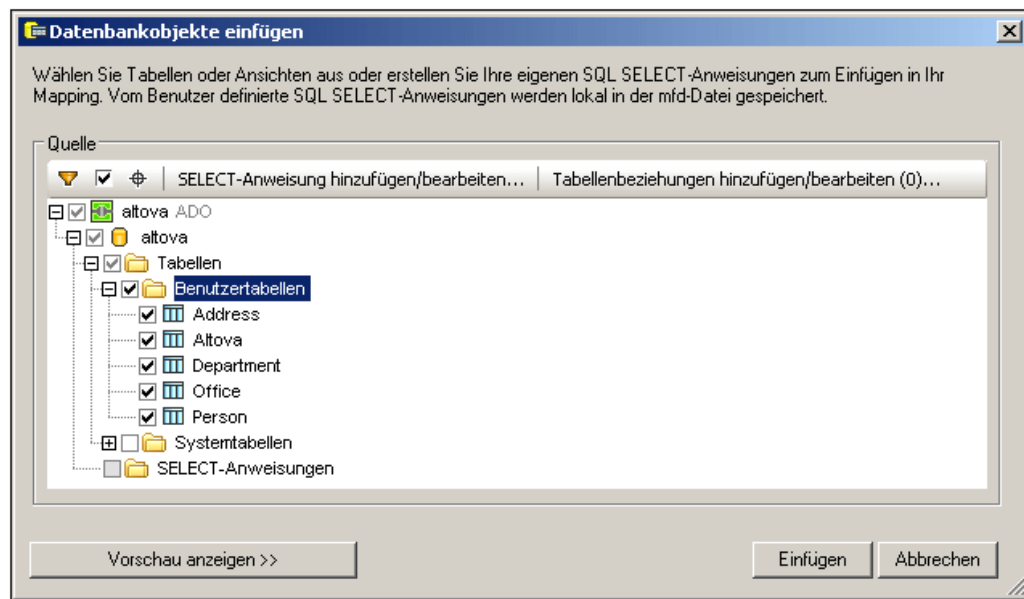
1. Klicken Sie in der Titelleiste auf das Programmiersprachensymbol (Java, C#, C++ oder BUILTIN) um die Sprache auszuwählen, die der erzeugte Code unterstützen soll. Damit wird auch die entsprechende Bibliothek im Bibliotheksfenster geladen.
2. Klicken Sie auf das Symbol **XML-Schema | Datei einfügen**  und wählen Sie die Datei **Altova_Hierarchical.xsd** aus dem Ordner MapForceExamples aus.
3. Klicken Sie im Meldungsfeld, das nun angezeigt wird auf die Schaltfläche **Durchsuchen...** und wählen Sie die Datei **Altova-cmpy.xml** als XML-Instanzdatei aus.
4. Klicken Sie im Mapping-Fenster in der Komponente "Altova_Hierarchical" auf den Eintrag **Altova** und drücken Sie auf der Zahlentastatur die *-Taste, um alle Elemente einzublenden. Passen Sie ggf. die Größe der Komponente an.
5. Klicken Sie auf das Symbol **Datenbank einfügen** , wählen Sie im Dialogfeld **Datenbank auswählen** das Symbol **Verbindungsassistent** aus.
6. Wählen Sie den Eintrag **Microsoft Access (ADO)** und klicken Sie auf **"Weiter"**.



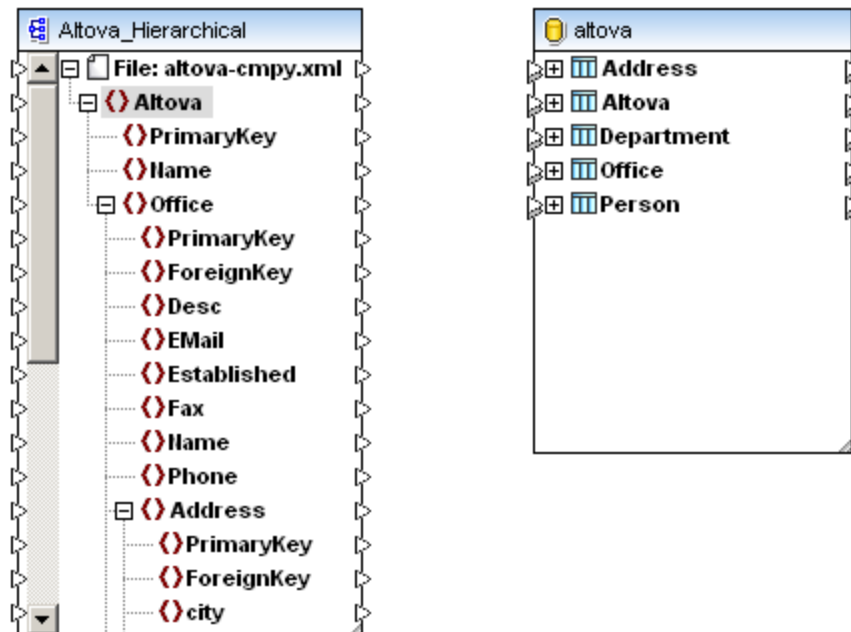
7. Klicken Sie auf die Schaltfläche "Durchsuchen", um im Ordner [...\MapForceExamples\Tutorial\](#) die Datenbank **altova.mdb** auszuwählen, und anschließend auf **Verbinden**.



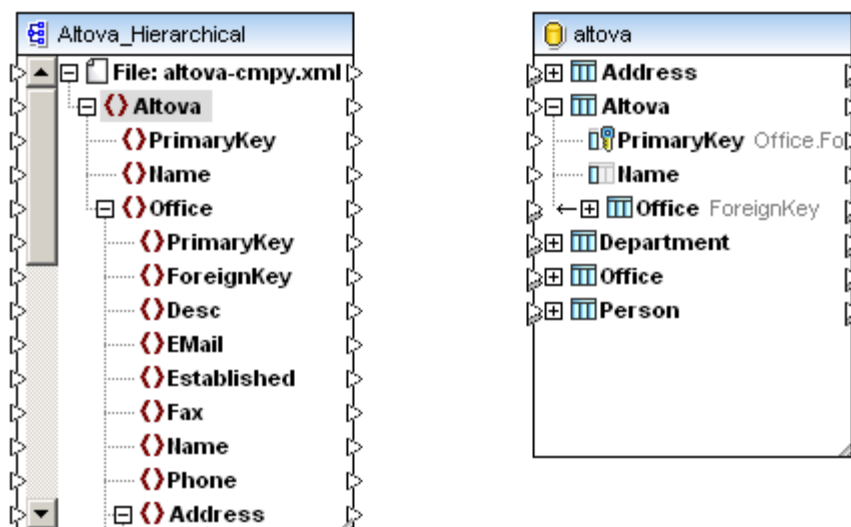
Im angezeigten Dialogfeld "Datenbankobjekte einfügen" können Sie definieren, welche Tabellen, Ansichten oder Systemtabellen in der Datenbankkomponente angezeigt werden sollen. Durch Klicken auf die Schaltfläche **Vorschau anzeigen** können Sie eine Vorschau der ausgewählten Tabelle anzeigen.



8. Klicken Sie auf das Kontrollkästchen links von Benutzertabellen, um alle auszuwählen und klicken Sie auf OK um die Datenbank einzufügen.



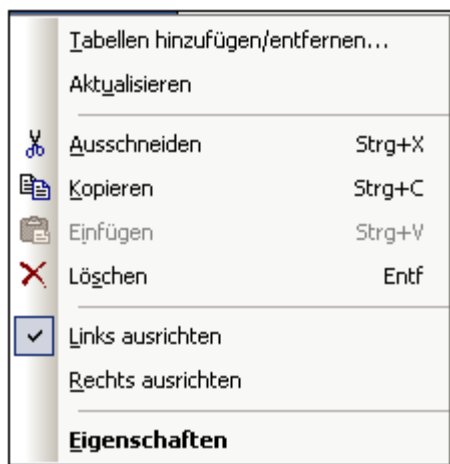
9. Klicken Sie auf das + Symbol des Datenelements **Altova**, um die Altova Tabellenfelder einzublenden.



Ändern der Datenbankeinstellungen

Die Datenbankeinstellungen können durch Rechtsklick auf die Datenbank und Auswahl der folgenden Befehle geändert werden:

- **Tabellen hinzufügen/entfernen:** Dient zum Hinzufügen oder Löschen von Tabellen zu der/aus der aktuellen Komponente
- **Eigenschaften:** Dient zum Ändern der Komponentendatenbank durch Klicken auf die Schaltfläche "Ändern" und Verwendung des Assistenten zum Auswählen einer anderen Datenbank.



17.2.2 Einfügen von Datenbanken - Anpassen der Tabellenvorschau

Das Dialogfeld **Datenbankobjekte einfügen** enthält eine Symbolleiste, über die Sie bestimmte Objekte im Gruppenfeld "Quelle" anpassen oder suchen können.



Mit Hilfe der **Objektsuche** können Sie bestimmte Datenbankobjekte suchen



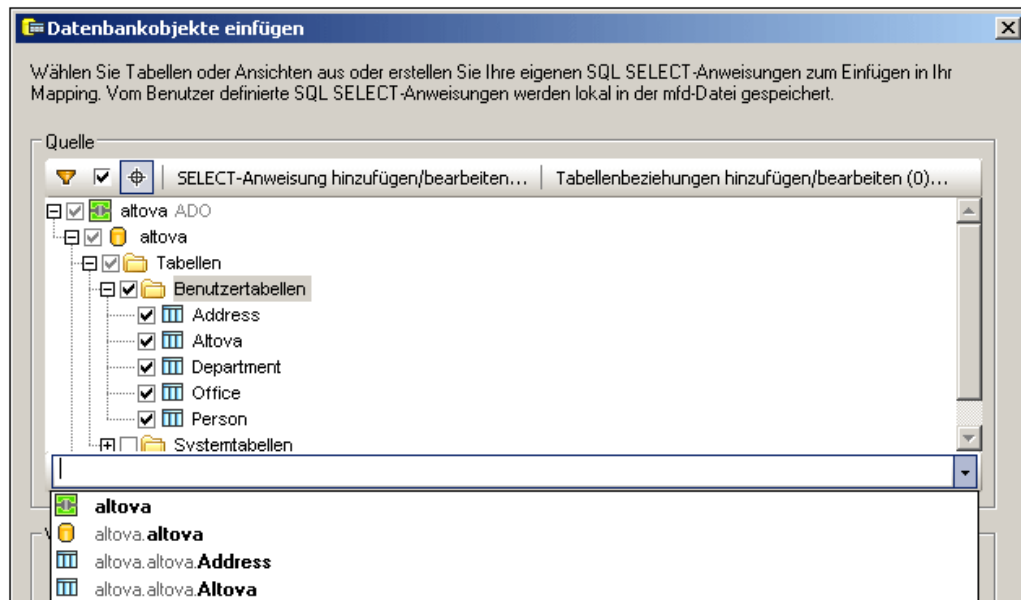
Mit Hilfe von **Filter** können Sie die Anzeige auf Tabellen mit bestimmten Zeichen einschränken



Mit Hilfe der Schaltfläche **Nur ausgewählte Objekte anzeigen** können Sie nur diejenigen Objekte anzeigen, deren Kontrollkästchen aktiviert ist

Suchen von Datenbankelementen über die Objektsuche:

1. Klicken Sie auf das Symbol **Objektsuche**  oder drücken Sie **Strg+L** um nach bestimmten Datenbankobjekten zu suchen. Daraufhin wird am unteren Rand des Fensters eine Dropdown-Liste mit allen Objekten angezeigt, die zur Auswahl stehen.

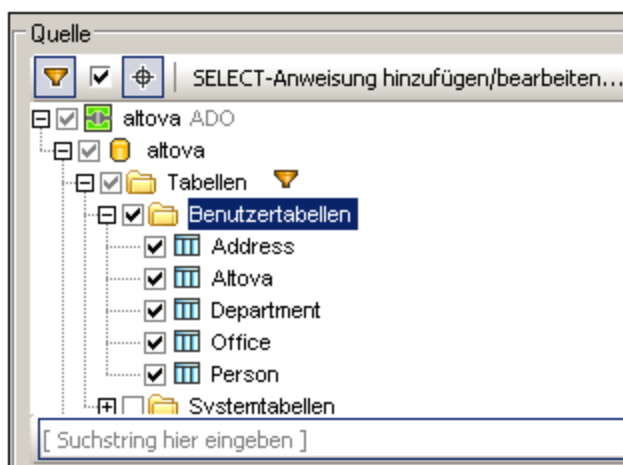


2. Geben Sie den String ein, nach dem Sie suchen möchten, oder wählen Sie ein Objekt aus der Dropdown-Liste aus.

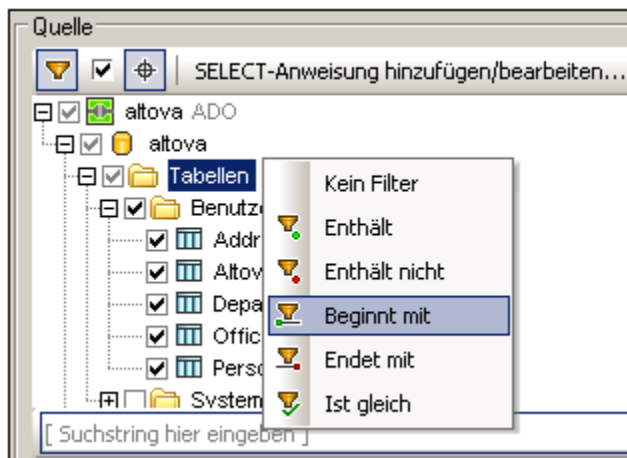
Filtern von Objekten im Vorschaufenster:

Schemas, Tabellen und Ansichten können nach dem Namen oder dem Teil eines Namens gefiltert werden. Die Groß- und Kleinschreibung wird dabei nicht beachtet.

1. Klicken Sie in der Symbolleiste auf das Symbol **Ordnerinhalt filtern** , um die Filterung zu aktivieren. Neben den Ordnern werden Filtersymbole angezeigt.

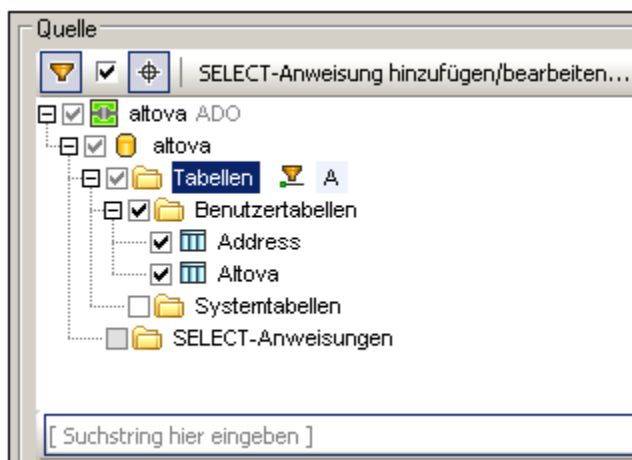


2. Klicken Sie auf das Filtersymbol neben dem Ordner und den Datenbankobjekten, die Sie filtern möchten. Wählen Sie im Popup-Menü, das daraufhin angezeigt wird, die Filteroption aus.



Neben dem Filtersymbol erscheint ein leeres Feld.

3. Geben Sie die Zeichen ein, anhand derer gefiltert werden soll, z.B. A.

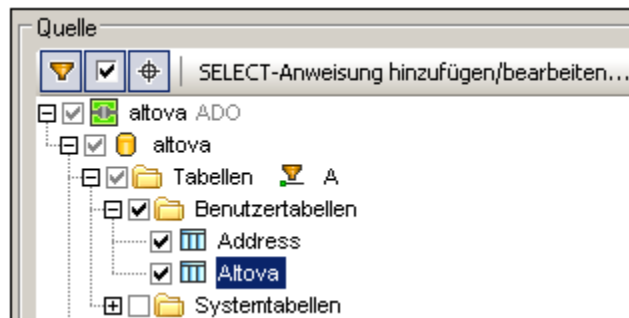


Die Ergebnisse werden während der Eingabe automatisch aktualisiert.

Anzeigen ausschließlich der Objekte, die mit einem Häkchen versehen sind

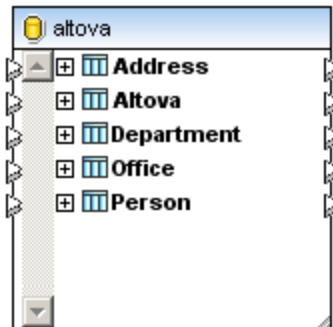
1. Klicken Sie in der Symbolleiste auf das Symbol **Nur ausgewählte Objekte anzeigen**

 , um nur die Tabellen anzuzeigen, die mit einem Häkchen markiert sind.



17.2.3 Komponenten und Tabellenbeziehungen

Tabellenbeziehungen sind in der Datenbankkomponente klar ersichtlich. In der Datenbankkomponente wird jede einzelne Tabelle einer Datenbank als "**Root**"-Tabelle in einer Baumstruktur mit den damit in Zusammenhang stehenden Tabellen unterhalb davon angezeigt.



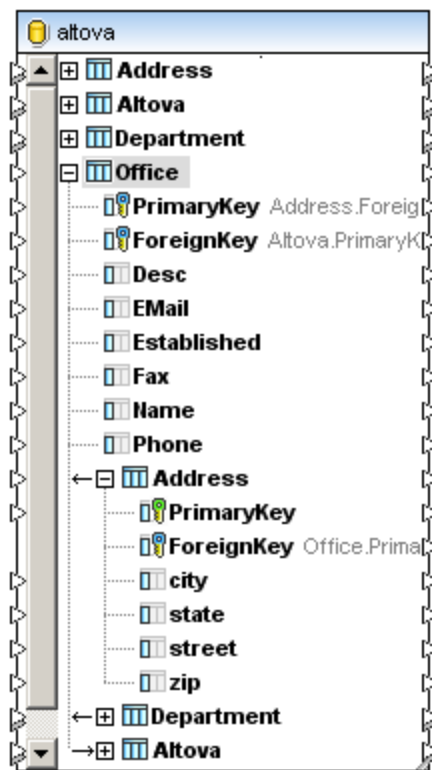
Die im obigen Diagramm angezeigten Tabellennamen bezeichnen wir als "**Root**"-Tabellen, d.h. sie befinden sich auf der obersten Ebene, und bilden sozusagen den Stamm oder die "Root" der Baumstruktur. Wenn man einen Tabellennamen erweitert, werden alle damit verknüpften Tabellen angezeigt. Die "Root"-Tabellen werden normalerweise in alphabetischer Reihenfolge aufgelistet, dies hat jedoch keinen Einfluss auf die tatsächlichen Tabellenbeziehungen.

Stellen Sie beim Erzeugen von Datenbankabfragen/Mappings mit Tabellenbeziehungen, auch bei SQL/XML-Datenbanken im flachen Format, sicher, dass Sie **zwischen Tabellen**, die unterhalb **einer der "Root"-Tabellen** angezeigt werden, Mappings erzeugen, wenn die Tabellenbeziehung nicht verloren gehen soll, d.h. wenn Sie Abfragen erstellen, die Joins (Verbindungen) verwenden.

In der unten gezeigten Abbildung sehen Sie die erweiterte "Root"-Tabelle **Office** der Altova-Datenbank. Die Pfeile links von den Erweiterungs-/Verkleinerungssymbolen vor den Tabellennamen und die Einrückungslinien zeigen die Tabellenbeziehungen an.

Von der Tabelle **Office** die Baumstruktur abwärts:

- Der **Pfeil nach links** zeigt an, dass es sich um eine Child-Tabelle der obigen Tabelle handelt, **Address** ist eine Child-Tabelle von Office.
- Department ist ebenfalls eine Child-Tabelle von Office und eine "Schwestertabelle" von Address, beide haben dieselbe Einrückungslinie.
- Person ist eine Child-Tabelle von Department.
- Der **Pfeil nach rechts** zeigt an, dass es sich um eine "Parent"-Tabelle (übergeordnete Tabelle) der obigen Tabelle handelt, **Altova** ist ein Parent der Tabelle Office.



Welche "Root"-Tabellen soll ich für das Mappen von Daten verwenden?

Stellen Sie beim Mappen auf Datenbanktabellen sicher, dass Sie dabei die **jeweils gewünschte** "Root"-Tabelle als oberste Ebene definieren.

Beispiel:

Angenommen, Sie möchten lediglich die Daten der Tabelle "Person" einfügen oder aktualisieren. Sie sollten dann Mappings mit der Tabelle "Person" als "Root"-Tabelle erzeugen und dann die Datenelemente der zu aktualisierenden Personenelemente aufeinander mappen.

Wenn Sie Daten aus den beiden Tabellen "Department" und "Person" aktualisieren möchten und die Datenbankbeziehung zwischen diesen Tabellen beibehalten werden soll, setzen Sie die Tabelle "Department" als "Root"-Tabelle und erzeugen Sie Mappings zwischen den Quell- und Zieldatenelementen der beiden Tabellen.

17.2.4 Datenbankaktion: Einfügen

In diesem Abschnitt des Tutorials wird erläutert, wie man Daten in eine Datenbank einfügt. Anhand der Quelldaten aus einer XML-Datei und des entsprechenden XML-Schemas werden SQL-Scripts generiert, die die entsprechenden Datenbankaktionen ausführen.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man mit MapForce Daten in eine Datenbank einfügt. Insbesondere lernen Sie:

- wie man ein neues Büroorganigramm (orgchart) zur Tabelle Altova in Ihrer Beispieldatenbank hinzufügt
- wie man damit in Zusammenhang stehende office-Tabellen zum neuen orgchart-Datensatz hinzufügt.

In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Datenbank einfügen: Dieser Befehl befindet sich in der Symbolleiste "Komponente einfügen" und im Menü **Einfügen**. Klicken Sie auf diesen Befehl, um ein Dialogfeld **Datenbank auswählen** zu öffnen, in dem Sie eine Datenbankverbindung definieren und auswählen können, welche Datenbanktabellen in das MapForce Mapping inkludiert werden sollen.



Datenbankaktionen: Dieser Befehl befindet sich im Menü **Komponente** einer Datenbankkomponente und wird in der Datenbankkomponente in Form einer Schaltfläche rechts vom Tabellennamen angezeigt. Klicken Sie auf diesen Befehl, um das Dialogfeld **Datenbankaktionen** aufzurufen, in dem Sie definieren können, welche Aktionen für die einzelnen Datensätze ausgeführt werden sollen.



SQL-Script ausführen: Dieser Befehl befindet sich in der Symbolleiste "Ausgabevorschau" und im Menü "Ausgabe". Klicken Sie auf diesen Befehl, um das im Ausgabefenster generierte SQL-Script auszuführen.



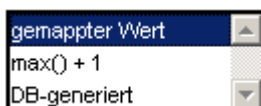
Ausgabedatei neu generieren: Dieser Befehl befindet sich in der Symbolleiste "Ausgabevorschau" und im Menü "Ausgabe". Klicken Sie auf diesen Befehl, um das SQL-Script zurückzusetzen, damit es ein zweites Mal ausgeführt werden kann.

Einfügen von Daten in eine Datenbanktabelle

Im ersten Beispiel dieses Abschnitts wird die einfache Aufgabe erklärt, wie man zu der Tabelle Altova ein neues Organigramm **hinzufügt**. Für jeden Datensatz wird ein neuer Primärschlüssel generiert.

Primärschlüsseleinstellungen

Die Primärschlüsseleinstellungen für die Einfügeaktion werden über die Auswahllisten rechts vom jeweiligen Feld definiert.



Es stehen die folgenden Optionen zur Auswahl:

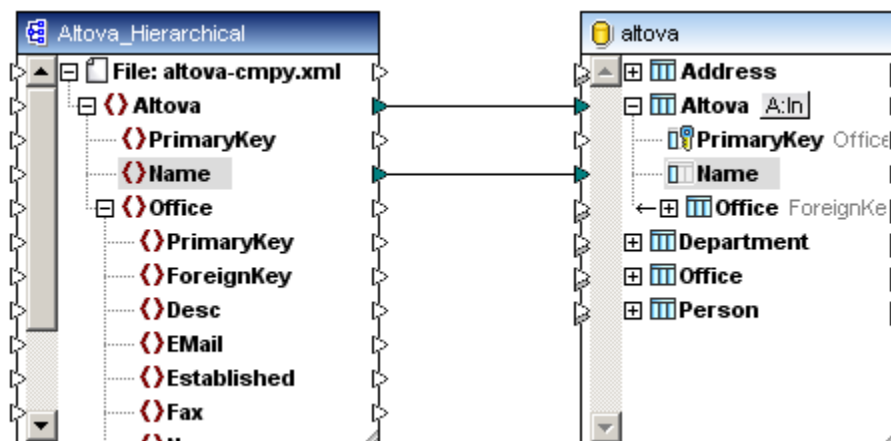
- Mit der Option **gemappter Wert** können Quelldaten direkt auf das Datenbankfeld gemappt werden. Dies ist die Standardeinstellung für alle Datenbankfelder.
- Um die Schlüsselwerte auf Basis der bestehenden Schlüssel in der Datenbank zu generieren, wählen Sie "**Max() + 1**", damit neue Datensätze automatisch an bestehende angehängt werden.
- Verwenden Sie die Einstellung **DB-generiert**, wenn die Datenbank die Schlüsselwerte generiert bzw. mit Hilfe der **Identity-Funktion** generiert.

So fügen Sie Werte aus einer XML-Datei in eine Datenbanktabelle ein:

1. Öffnen Sie ein neues Mapping-Fenster und fügen Sie das Schema **Altova_Hierarchical.xsd** ein (und weisen Sie ihm **altova-cmpy.xml** als XML-Eingabeinstanz zu). Fügen Sie die Datenbank altova.mdb, wie im Abschnitt [Einrichten von XML-auf-Datenbank Mappings](#) beschrieben, in das Mapping-Fenster ein.
2. Erzeugen Sie die folgenden Mappings:

Altova auf Altova, und
Name auf Name

Bitte beachten Sie: Wenn alle Altova-, Office- usw. Datenelemente automatisch gemappt werden, ist die Option "Identische Sub-Einträge automatisch verbinden" aktiv. Wählen Sie den Befehl **Bearbeiten | Rückgängig** und anschließend die Menüoption **Verbindung | Identische Sub-Einträge automatisch verbinden**, um die Option zu deaktivieren.



4. Das Symbol rechts von der Tabelle "Altova" (In der Abbildung oben **A:In**) zeigt die derzeit für diese Tabelle definierte Tabellenaktion an, d.h. Aktion:Einfügen (In = Insert). Wenn Sie auf das Symbol klicken, wird das Dialogfeld **Datenbankaktionen** geöffnet, in dem verschiedenen Tabellenaktionen definiert werden können.
5. Klicken Sie auf das Tabellenaktionssymbol **A:In** neben dem Tabelleneintrag **Altova**. In diesem Dialogfeld ist im Moment nur eine Tabellen-Aktionsspalte definiert, nämlich **Alles einfügen**.

Datenbankaktionen - Altova

Vor dem ersten Datensatz auszuführende SQL-Anweisung

☒ Keine
☐ Alle Datensätze LÖSCHEN ☐ auch alle Datensätze in Child-Tabellen löschen
☐ Benutzerdefinierte SQL:

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle Daten überein, wird die jeweilige Aktion ausgeführt.

Aktion an Input-Daten	gemappter Wert
PrimaryKey	gemappter Wert
xmlns	gemappter Wert
ipo	gemappter Wert
xsi	gemappter Wert
schemaLocation	gemappter Wert
Name	gemappter Wert

Daten in Child-Tabellen löschen ☐
Input-Child-Daten ignorieren ☐
Office

☐ Transaktionen verwenden

Aktion anhängen
Aktion einfügen
Aktion löschen

OK Abbrechen

Die Tabellenaktion "Alle einfügen", fügt Datensätze mit allen **gemappten Feldern** der aktuellen Tabelle in die Datenbank ein. Wir müssen nun definieren, wie der neue Primärschlüssel (PrimaryKey) für diese Aktion generiert wird.

5. Klicken Sie in die Auswahlliste rechts vom Feld **PrimaryKey** und wählen Sie **max() + 1**.

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle Daten überein, wird die jeweilige Aktion ausgeführt.

Aktion an Input-Daten	gemappter Wert
PrimaryKey	max() + 1
Name	gemappter Wert

Daten in Child-Tabellen löschen ☐
Input-Child-Daten ignorieren ☐
Office

Aktion anhängen
Aktion einfügen

6. Bestätigen Sie die aktuellen Einstellungen mit OK.
7. Klicken Sie auf das Ausgaberegister am unteren Rand des Mapping-Fensters um das SQL Script zu sehen, das von diesem Mapping erzeugt wird.

```

/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
Provider=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Documents and Settings\... \My Documents\
Altova\MapForce2012\MapForceExamples\altova.mdb;
*/

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')

```

Das Script führt das Mapping auf die Zieldatenbank aus und berücksichtigt dabei die definierten Tabellenaktionen. Sie können SQL Scripts über das Ausgabefenster erneut ausführen, indem Sie auf die Schaltfläche **Ausgabedatei neu generieren**  klicken.

7. Klicken Sie in der Funktionsleiste auf das Symbol **SQL-Script ausführen** , um das Script auszuführen und die Tabellendaten in die Datenbank einzufügen. Nach erfolgreichem Abschluss des Vorgangs wird im Fenster "Meldungen" eine Bestätigungsmeldung angezeigt. Bestätigen Sie mit OK.
8. Öffnen Sie die Altova-Datenbank in DatabaseSpy oder Access, um zu sehen, wie sich diese Aktion ausgewirkt hat.

Altova : Table		
	PrimaryKey	Name
+	1	Organization Chart
+	2	Microtech OrgChart
▶		

Der Altova-Tabelle wurde ein neuer Microtech OrgChart-Datensatz mit dem neuen PrimaryKey 2 hinzugefügt. Die Daten für diesen Datensatz stammen aus der XML-Eingabeinstanz.

9. Wechseln Sie wieder zurück zu MapForce.

Im Fenster "Meldungen" sehen Sie nun eine Aufzeichnung darüber, was bei der Verarbeitung des SQL-Scripts passiert ist.

```

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> OK. One or more rows.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 row(s).

```

Einfügen von Tabellen und damit in Zusammenhang stehenden untergeordneten Tabellen

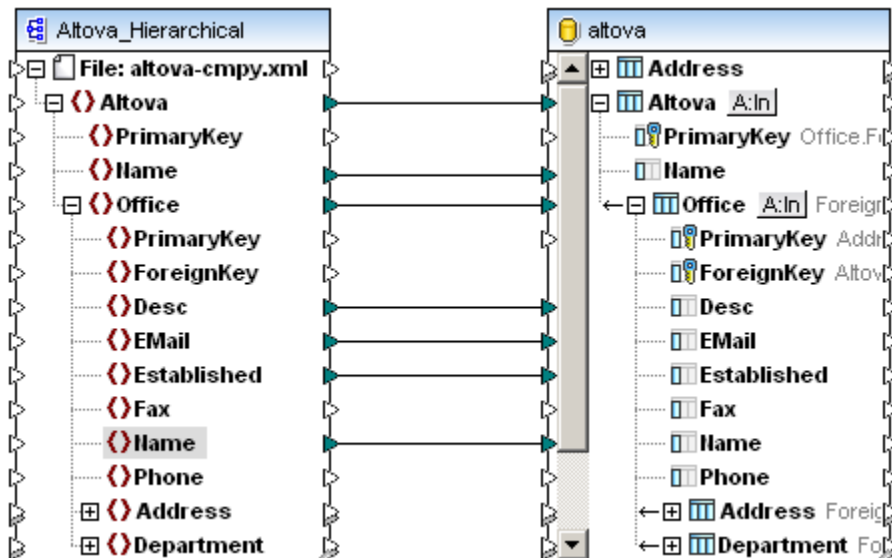
Als Ausgangsbasis für dieses Beispiel wird das [vorhergehende Beispiel](#) verwendet. Es wird durch Einfügen von untergeordneten Office-Tabellen in die übergeordnete Altova-Tabelle erweitert.

Tabellenbeziehungen werden nur dann automatisch generiert, wenn Mappings zwischen untergeordneten Tabellen einer "Root"-Tabelle erstellt werden. In diesem Fall werden Mappings zwischen den Office-Feldern, die sich direkt unterhalb der Altova-Parent (oder "Root")-Tabelle

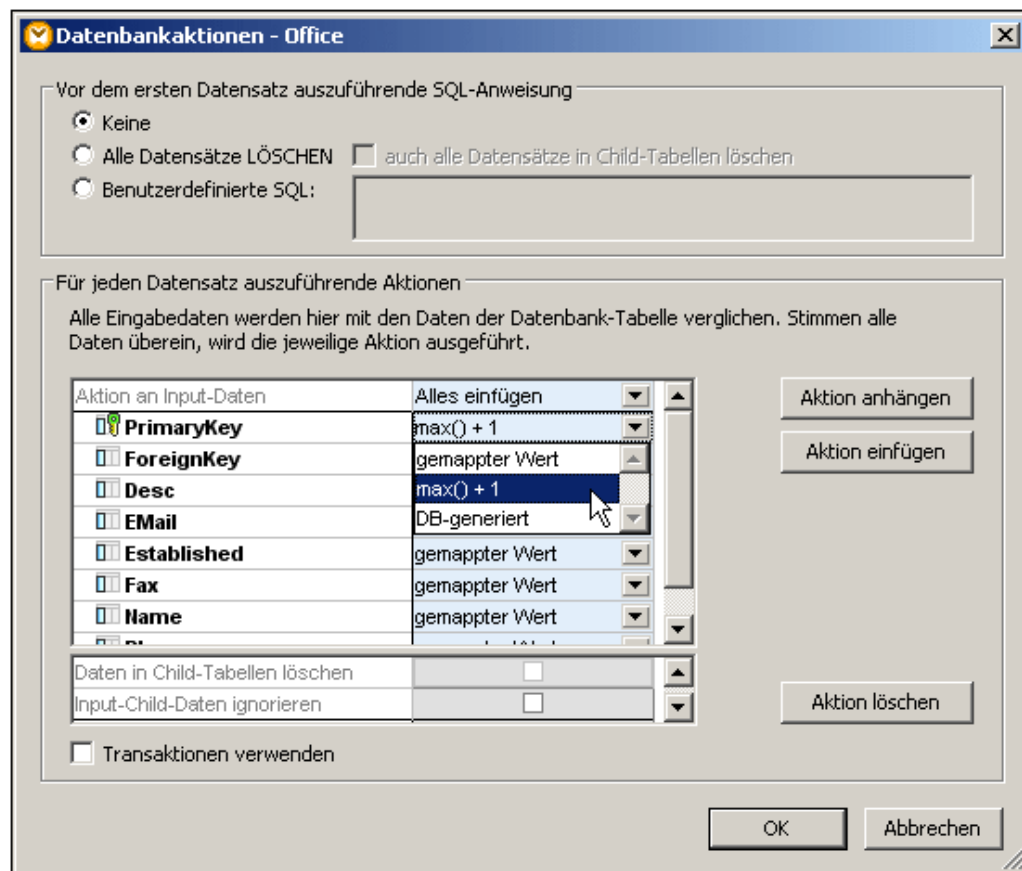
befinden, erzeugt.

Einfügen von Tabellen und verwandten Child-Tabellen:

1. Öffnen Sie ein neues Mapping-Fenster und fügen Sie das Schema **Altova_Hierarchical.xsd** ein (und weisen Sie ihm **altova-cmpy.xml** als XML-Eingabeinstanz zu). Fügen Sie die Datenbank **altova.mdb**, wie im Abschnitt [Einrichten von XML-auf-Datenbank Mappings](#) beschrieben, in das Mapping-Fenster ein.
2. Erzeugen Sie Mappings zwischen den Datenelementen Altova und Name, wie im [vorherigen Abschnitt](#) beschrieben.
3. Klicken Sie in beiden Komponenten auf das +-Erweiterungssymbol des Datenelements **Office**, um die Office-Tabellenfelder anzuzeigen.
4. Erzeugen Sie zwischen den beiden Komponenten die folgenden Mappings:
Office auf Office
Desc auf Desc
EMail auf EMail
Established auf Established, und
Name auf Name.



5. Klicken Sie auf das **A:In** Tabellenaktionssymbol neben dem Tabelleneintrag **Office**. Die Tabellenaktion **Alles einfügen..** ist standardmäßig ausgewählt. Sie müssen hier keine Änderungen vornehmen.
6. Klicken Sie in die **PrimaryKey**-Auswahlliste und wählen Sie den Eintrag **max()+1** und klicken Sie zur Bestätigung auf OK.



7. Klicken Sie auf die Schaltfläche Ausgabe, um das SQL-Skript anzuzeigen.

```

/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
Provider=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Documents and Settings\...My Documents\Altova\
MapForce2012\MapForceExamples\altova.mdb;
*/

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM
[Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM
[Office]
-->>> %PrimaryKey2%

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (
'%PrimaryKey1%', 'Microtechnology products are currently the bleeding edge of computer technology.',
'office@microtech.com', '1992-04-01', 'Microtech, Inc.', '%PrimaryKey2%')

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM
[Office]
-->>> %PrimaryKey3%

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (
'%PrimaryKey1%', 'Microtech established its new office on Feb 30', 'nextoffice@microtech.com',
'2001-03-01', 'Microtech Partners, Inc.', '%PrimaryKey3%')

```

8. Klicken Sie auf das Symbol "SQL-Skript ausführen" , um das Skript zu starten und die neuen Tabellen einzufügen.
9. Öffnen Sie die Datenbank in MS Access und doppelklicken Sie auf die Tabelle **Altova**, um das Ergebnis in MS-Access zu sehen.

Altova : Table							
		PrimaryKey	Name				
	-	1 Organization Chart					
			PrimaryKey	Desc	EEmail	Established	
		+	1	The company was es	office@nanonul	1992-04-01	
		+	2	On March 1st, 2000,	nextoffice@nan	2001-03-01	
		*					
▶	-	2 Microtech OrgChart					
				PrimaryKey	Desc	EEmail	Established
		+	3	Microtechnology proc	office@microtec	1992-04-01	
		▶	+	4	Microtech establishe	nextoffice@mic	2001-03-01
		*					
	*						

- Dem Microtech OrgChart wurden zwei neue Büros (Office) hinzugefügt.
10. Doppelklicken Sie auf die Tabelle **Office**, um die Einzelheiten des erzeugten Ergebnisses zu sehen.

Office : Table				
	PrimaryKey	ForeignKey	Desc	Email
▶ +	1	1	The company was establis	office@nanonul
+	2	1	On March 1st, 2000, Nanoi	nextoffice@nan
+	3	2	Microtechnology products	office@microtec
+	4	2	Microtech established its r	nextoffice@mic
*				

Die neuen Büros wurden mit den Primärschlüsseln 3 bzw. 4 hinzugefügt. Beide Büros sind über den Sekundärschlüssel 2 mit der Tabelle Altova verknüpft, der auf den Datensatz Microtech OrgChart referenziert.

```

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
-->>> OK. One or more rows.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 row(s).

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
-->>> OK. One or more rows.

INSERT INTO [Office] ([ForeignKey], [Desc], [Email], [Established], [Name], [PrimaryKey]) VALUES (2,
'Microtechnology products are currently the bleeding edge of computer technology.', 'office@microtech.com',
'1992-04-01', 'Microtech, Inc.', 3)
-->>> OK. 1 row(s).

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
-->>> OK. One or more rows.

INSERT INTO [Office] ([ForeignKey], [Desc], [Email], [Established], [Name], [PrimaryKey]) VALUES (2, 'Microtech
established its new office on Feb 30', 'nextoffice@microtech.com', '2001-03-01', 'Microtech Partners, Inc.', 4)
-->>> OK. 1 row(s).

```

17.2.5 Datenbankaktion: Aktualisieren

In diesem Abschnitt wird erklärt, wie man Datenbanken aktualisiert. Die verschiedenen Datenbankaktionen, die MapForce zu diesem Zweck zur Verfügung stellt, werden beschrieben.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie Sie mit Hilfe von MapForce Daten in einer Datenbanktabelle aktualisieren können. Insbesondere werden die folgenden Schritte beschrieben:

- [Aktualisieren der Fleder einer bestimmten Tabelle](#)
- [Hinzufügen neuer Datensätze zu einer Tabelle](#)
- [Verwendung der Aktualisieren, wenn... Bedingung](#)

In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Datenbank einfügen: Dieser Befehl befindet sich in der Symbolleiste "Komponente einfügen" und im Menü **Einfügen**. Klicken Sie auf diesen Befehl, um ein Dialogfeld **Datenbank auswählen** zu öffnen, in dem Sie eine Datenbankverbindung definieren und auswählen können, welche Datenbanktabellen in das MapForce Mapping inkludiert werden sollen.



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identischer Sub-Einträge ein- bzw. auszuschalten.



Datenbankaktionen: Dieser Befehl befindet sich im Menü **Komponente** einer Datenbankkomponente und wird in der Datenbankkomponente in Form einer Schaltfläche rechts vom Tabellennamen angezeigt. Klicken Sie auf diesen Befehl um das Dialogfeld **Datenbankaktionen** aufzurufen, in dem Sie definieren können, welche Aktionen für die einzelnen Datensätze ausgeführt werden sollen.



SQL-Script ausführen: Dieser Befehl befindet sich in der Symbolleiste "Ausgabevorschau" und im Menü "Ausgabe". Klicken Sie auf diesen Befehl, um das im Ausgabefenster generierte SQL-Script auszuführen.

In diesem Beispiel verwendete Dateien

- Altova_Hierarchical.xsd
- altova-cmpy.xml
- altova-cmpy-extra.xml
- altova.mdb

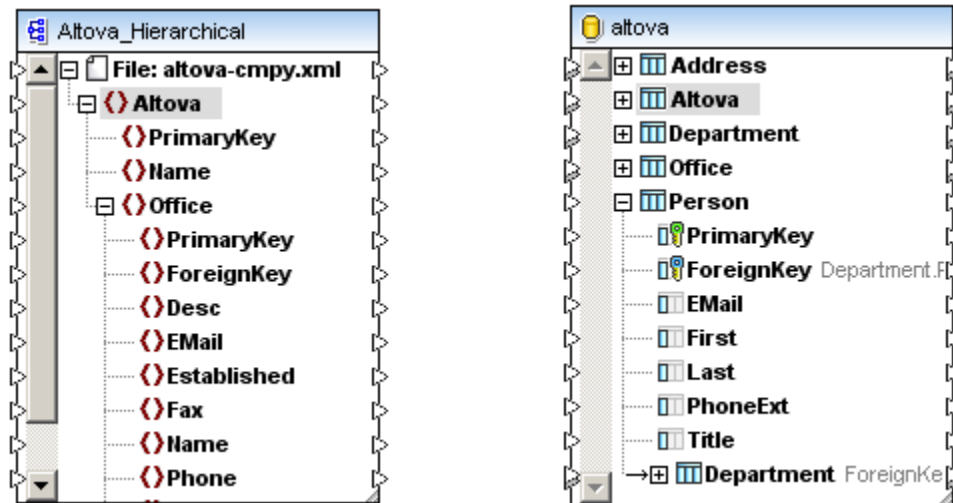
Aktualisieren von Datenbankfeldern

Im ersten Beispiel dieses Abschnitts wird die einfache Aufgabe erklärt, wie man bestehende Personendatensätze aktualisiert. Es werden Mappings von der XML-Datenquelle zur "Root"-Tabelle Person erzeugt.

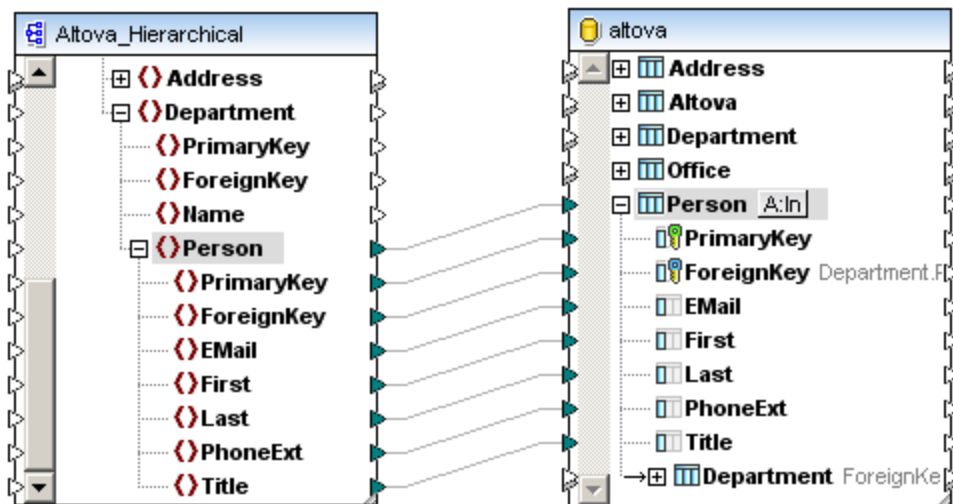
So aktualisieren Sie die Tabelle "Person":

1. Fügen Sie das Schema Altova_Hierarchical.xsd ein (und weisen Sie ihm

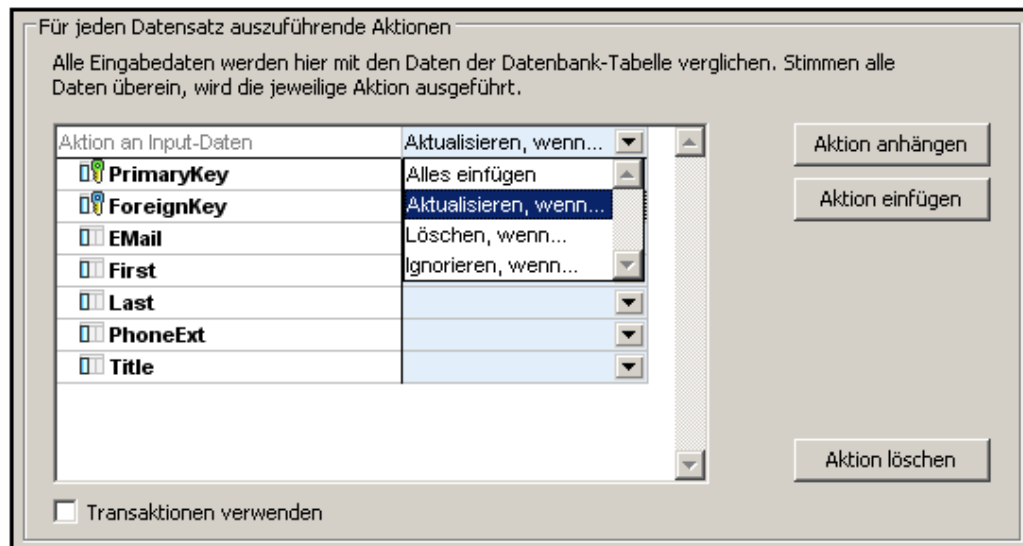
- altova-cmpy.xml** als XML-Eingabeinstanz zu), wie im [vorigen Abschnitt](#) beschrieben.
- Fügen Sie die MS Access-Datenbank **altova.mdb** in das Mapping ein, wie im [vorigen Abschnitt](#) beschrieben.



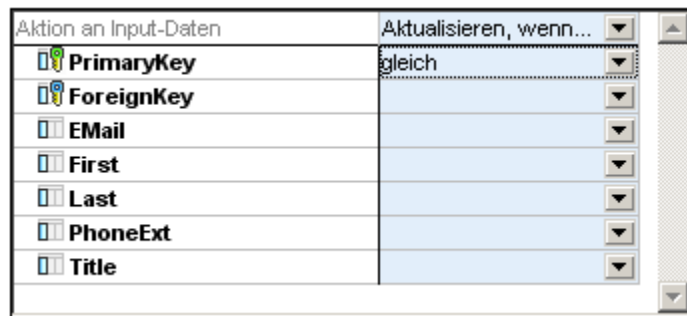
- Aktivieren Sie die Schaltfläche **Identische Sub-Einträge automatisch verbinden**.
- Erweitern Sie in der XML-Schema-Komponente die Datenelemente Office und Department, und klicken Sie auf das Element **Person** und ziehen Sie den Konnektor zum Datenelement "Person" der Datenbank. Stellen Sie sicher, dass Sie es mit der "Root"-Tabelle "Person" verbinden. Alle übereinstimmenden Child-Elemente werden automatisch gemappt.



- Klicken Sie auf das **Person**-Tabellenaktionssymbol **A:In**, um das Dialogfeld zu öffnen.
- Klicken Sie in der Auswahlliste "Aktion an Input-Daten" auf den obersten Eintrag und wählen Sie **Aktualisieren, wenn...**

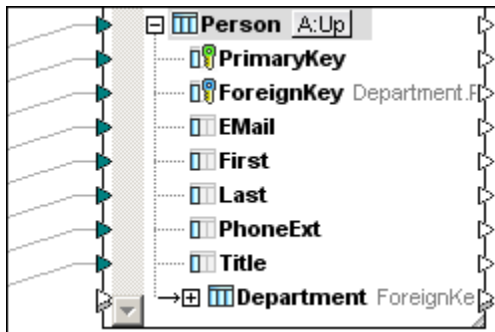


7. Klicken Sie auf das Dropdown-Listenfeld in der Zeile des Eintrags PrimaryKey, wählen Sie den Eintrag **gleich** und bestätigen Sie mit **OK**.



Die Primärschlüssel von Quelle und Ziel werden mit Hilfe des Operators "gleich" verglichen. Wenn die beiden Schlüssel identisch sind, werden alle gemappten Felder der Personentabellen aktualisiert.

Das Tabellenaktionssymbol hat sich jetzt in **A:Up** geändert, um anzuzeigen, dass die Tabellenaktion "Aktualisieren" (Update) ausgewählt wurde.



8. Klicken Sie auf die Schaltfläche Ausgabe am unteren Rand des Mapping-Fensters, um das von diesem Mapping erzeugte SQL-Skript anzuzeigen. Das Skript führt das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.
9. Klicken Sie in der Funktionsleiste auf die Schaltfläche **SQL-Skript ausführen** , um

das Script auszuführen und die Datensätze der Datenbank zu aktualisieren. Bei erfolgreicher Ausführung erscheint eine Bestätigungsmeldung im Fenster "Meldungen".

10. Öffnen Sie die **Altova**-Datenbank in MS Access und doppelklicken Sie auf die Tabelle **Person**, um das Ergebnis zu sehen.

Alle Personendatensätze der Datenbank wurden aktualisiert.

Person : Table						
	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt
	1	1	A.Aldrich@micr	Albert	Aldrich	582
	2	1	b.bander@micr	Bert	Bander	471
	3	1	c.clovis@micr	Clive	Clovis	963
	4	2	d.Durnell@micr	Dave	Durnell	621
	5	2	e.ellas@microt	Eve	Ellas	753
	6	3	f.fortunas@micr	Fred	Fortunas	951
	7	3	g.gundall@micr	Gerry	Gundall	654
	8	3	h.hardy@micr	Harry	Hardy	852
	9	3	i.idilko@microte	Ingrid	Idilko	951
	10	3	j.judy@microte	June	Judy	753
	11	3	k.krove@microt	Karl	Krove	334

Aktualisieren und Hinzufügen von neuen Datensätzen

In diesem etwas komplexeren Beispiel wird versucht, Datensätze sowohl in der Tabelle "Department" als auch in der Tabelle "Person" zu aktualisieren und außerdem alle neuen Personendatensätze, die die XML-Eingabedatei enthält, hinzuzufügen. Als "**Root**"-Tabelle wird in diesem Beispiel daher die Tabelle **Department** verwendet.

In diesem Beispiel verwendete Dateien:

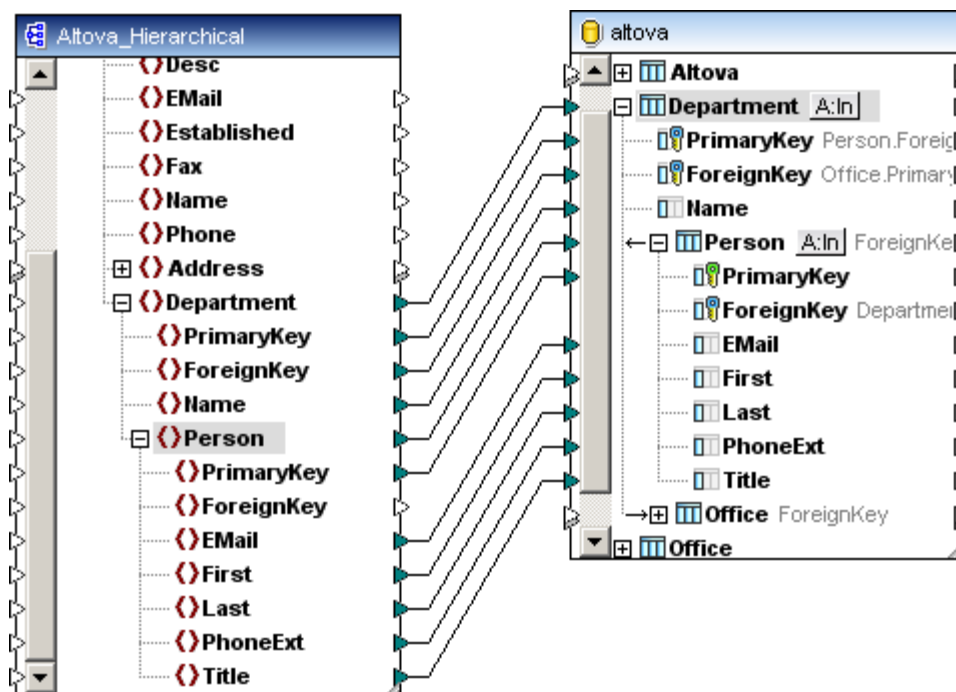
- Altova_Hierarchical.xsd
- altova-cmpy-**extra**.xml (die XML-Instanzdatei für Altova_hierarchical.xsd)
- altova.mdb

Ziel:

- **Aktualisierung** der Datensätze "Name" der Tabelle "Department"
- **Aktualisierung** vorhandener Personendatensätze
- **Einfügen** aller neuen Personendatensätze

So mappen Sie die Tabellen Department und Person aufeinander:

1. Fügen Sie das Schema Altova_Hierarchical.xsd ein (und weisen Sie ihm **altova-cmpy.xml** als XML-Eingabeinstanz zu), wie im [vorigen Abschnitt](#) beschrieben.
2. Fügen Sie die MS Access-Datenbank **altova.mdb** in das Mapping ein, wie im [vorigen Abschnitt](#) beschrieben.
3. Aktivieren Sie die Schaltfläche **Identifizieren Sub-Einträge automatisch verbinden** .
4. Erweitern Sie in der Datenbankkomponente die Datenelemente "Department" und "Person".
5. Erweitern Sie in der XML-Schema-Komponente die Datenelemente "Office" und "Department". Klicken Sie auf das Datenelement **Person** und ziehen Sie den Konnektor auf das Datenelement "Person" der Datenbank. Stellen Sie sicher, dass Sie es mit der Tabelle "Person" innerhalb der Tabelle "Department" verbinden. Alle übereinstimmenden untergeordneten Datenelemente werden automatisch verbunden.



Definieren der Tabellenaktionen

Die Quell- und Zielprimärschlüssel der beiden Tabellen werden mit Hilfe des Operators "gleich" verglichen. Sind die beiden Schlüssel identisch, werden die **gemappten** Felder der Tabellen "Department" und "Person" aktualisiert. Wenn der Vergleich (in der Tabelle Person) fehlschlägt, wird die nächste Tabellenaktion durchgeführt, d.h. "Rest einfügen".

So definieren Sie die Tabellenaktion für die Tabelle "Department":

1. Klicken Sie auf das **A:In** Tabellenaktionssymbol von **Person**, um das Dialogfeld zu öffnen.
2. Klicken Sie in der Auswahlliste "Aktion an Input-Daten" auf den obersten Eintrag und wählen Sie **Aktualisieren, wenn...**
3. Klicken Sie in die Auswahlliste in der Zeile mit dem Eintrag PrimaryKey, wählen Sie den Eintrag **gleich** und klicken Sie zur Bestätigung auf **OK**.

Datenbankaktionen - Department

Vor dem ersten Datensatz auszuführende SQL-Anweisung

☒ Keine

☐ Alle Datensätze LÖSCHEN ☐ auch alle Datensätze in Child-Tabellen löschen

☐ Benutzerdefinierte SQL:

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle Daten überein, wird die jeweilige Aktion ausgeführt.

Aktion an Input-Daten	Aktualisieren, wenn...
PrimaryKey	gleich
ForeignKey	
Name	

Daten in Child-Tabellen löschen ☒

Input-Child-Daten ignorieren ☐

Person	

☐ Transaktionen verwenden

Aktion anhängen

Aktion einfügen

Aktion löschen

OK Abbrechen

Die Primärschlüssel der Quell- und Zielkomponente werden mit Hilfe des "gleich" Operators verglichen. Wenn die beiden Schlüssel identisch sind, werden alle gemappten Felder der Department-Tabellen aktualisiert.

So definieren Sie die Tabellenaktion für die Tabelle "Person":

1. Klicken Sie auf das **A:In** Tabellenaktionssymbol von **Department**, um das Dialogfeld zu öffnen.
2. Klicken Sie in der Auswahlliste "Aktion an Input-Daten" auf den obersten Eintrag und wählen Sie **Aktualisieren, wenn...**
3. Klicken Sie in die Auswahlliste in der Zeile mit dem Eintrag **PrimaryKey** und wählen Sie den Eintrag **gleich**.

Datenbankaktionen - Person

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle Daten überein, wird die jeweilige Aktion ausgeführt.

Aktion an Input-Daten	Aktualisieren, wenn...
PrimaryKey	gleich
ForeignKey	Sekundärschlüssel
Email	
First	

Aktion anhängen

Aktion einfügen

- Klicken Sie auf die Schaltfläche **Aktion anhängen**, um eine neue

Tabellenaktionsspalte anzuhängen.

Die Tabellenaktion **Rest einfügen** wird automatisch als zweite Tabellenaktion eingefügt. Wenn die Aktion "Aktualisieren, wenn.." fehlschlägt, dann wird diese Tabellenaktion verarbeitet.

Aktion an Input-Daten	Aktualisieren, wenn...	Rest einfügen
PrimaryKey	gleich	gemappter Wert
ForeignKey	Sekundärschlüssel	Sekundärschlüssel
EMail		gemappter Wert
First		gemappter Wert
Last		gemappter Wert
PhoneExt		gemappter Wert
Title		gemappter Wert

☐ Transaktionen verwenden

- Klicken Sie auf OK, um die Definition der Tabellenaktionen fertig zu stellen. Beachten Sie, dass das Tabellenaktionssymbol nun zu **A:Up,In** aktualisiert wurde.



- Klicken Sie auf die Schaltfläche "Ausgabe", um das SQL Script zu sehen.

```
UPDATE [Department] SET [ForeignKey] = 1, [Name] = 'Admin' WHERE ([Department].[PrimaryKey]=1)

SELECT [ForeignKey], [PrimaryKey] FROM [Department] WHERE ([PrimaryKey]=1)
-->>> %ForeignKey1%

-->>> %PrimaryKey1%

DELETE FROM [Person] WHERE EXISTS(SELECT * FROM [Department] WHERE [Department].[PrimaryKey] =
[Person].[ForeignKey] AND ([Department].[PrimaryKey]='%PrimaryKey1%'))

UPDATE [Person] SET [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich', [PhoneExt] = 582
, [Title] = 'Manager' WHERE ([Person].[ForeignKey] = '%PrimaryKey1%') AND ([Person].[PrimaryKey]=1)
```

Bitte beachten Sie:

Das Script führt das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.

Verarbeitungsreihenfolge

Die Bedingung **Aktualisieren, wenn...** überprüft, ob die Primärschlüssel des Quell- und Zieldatenelements identisch sind.

Tabelle "Department"

- Wenn die **Bedingung "true"** ist, wird jeder Department-Datensatz, in dem die Schlüssel identisch sind, aktualisiert. Wenn es in der Datenbank Datensätze gibt, die in der Quelldatei keine Entsprechung haben, werden diese Datensätze beibehalten und bleiben unverändert (in diesem Beispiel die Tabelle Engineering).
- Wenn die **Bedingung "false" ist**, d.h. wenn Schlüssel in der Quellkomponente vorhanden sind, die in der Zieldatenbank keine Entsprechung haben, wird keiner der Department-Datensätze aktualisiert.

Tabelle "Person":

- Wenn die **Bedingung "true"** ist, wird jeder Person-Datensatz, in dem die Schlüssel identisch sind, aktualisiert. Wenn es in der Datenbank Datensätze gibt, die in der Quelldatei keine Entsprechung haben, werden diese Datensätze beibehalten und bleiben unverändert.
- Wenn die **Bedingung "false" ist**, d.h. wenn Schlüssel in der Quellkomponente vorhanden sind, die in der Zieldatenbank keine Entsprechung haben, wird zur nächsten Tabellenaktionsspalte weitergegangen: **Rest einfügen**...Mit dieser Aktion werden neue Person-Datensätze, falls vorhanden, in die Tabelle "Person" eingefügt. In diesem Beispiel werden der Abteilung Admin zwei neue Personendatensätze mit den Primärschlüsseln 30 bzw. 31 hinzugefügt.

Microsoft Access - [Altova : Table]					
File Edit View Insert Format Records Tools Window Help					

Aktualisieren und Löschen von Child-Daten

In diesem Abschnitt wird beschrieben, welche Auswirkung die Bedingung **Aktualisieren, wenn...** in Kombination mit den einzelnen möglichen Tabellenaktionen (Alles Einfügen,

Aktualisieren, wenn..., Löschen, wenn...), die für die verwandten Child-Tabellen definiert wurden, auf eine Parent-Tabelle hat. In allen Beispielen mit Ausnahme von einem ist die Option **"Daten in Child-Tabellen löschen"** aktiv. Sie können das Mapping aus dem vorigen Abschnitt in diesem Abschnitt weiter verwenden.

Für dieses Beispiel verwendete Dateien:

- Altova_hierarchical.xsd
- Altova-cmpy-extra.xml
- Altova.mdb

Die Einstellungen der übergeordneten Tabelle (Department) für Datenbankaktionen sind die folgenden:

Aktion an Input-Daten	Aktualisieren, wenn...
PrimaryKey	gleich
ForeignKey	
Name	
Daten in Child-Tabellen löschen	☒ *
Input-Child-Daten ignorieren	☐

* Bei der Aktion "Löschen, wenn..." kann dieses Kontrollkästchen auch deaktiviert werden (siehe Tabelle unten).

Das Ergebnis des Mappings hängt, davon ab, welche Aktion an den Input-Daten Sie für die **Child-Tabelle (Person)** auswählen:

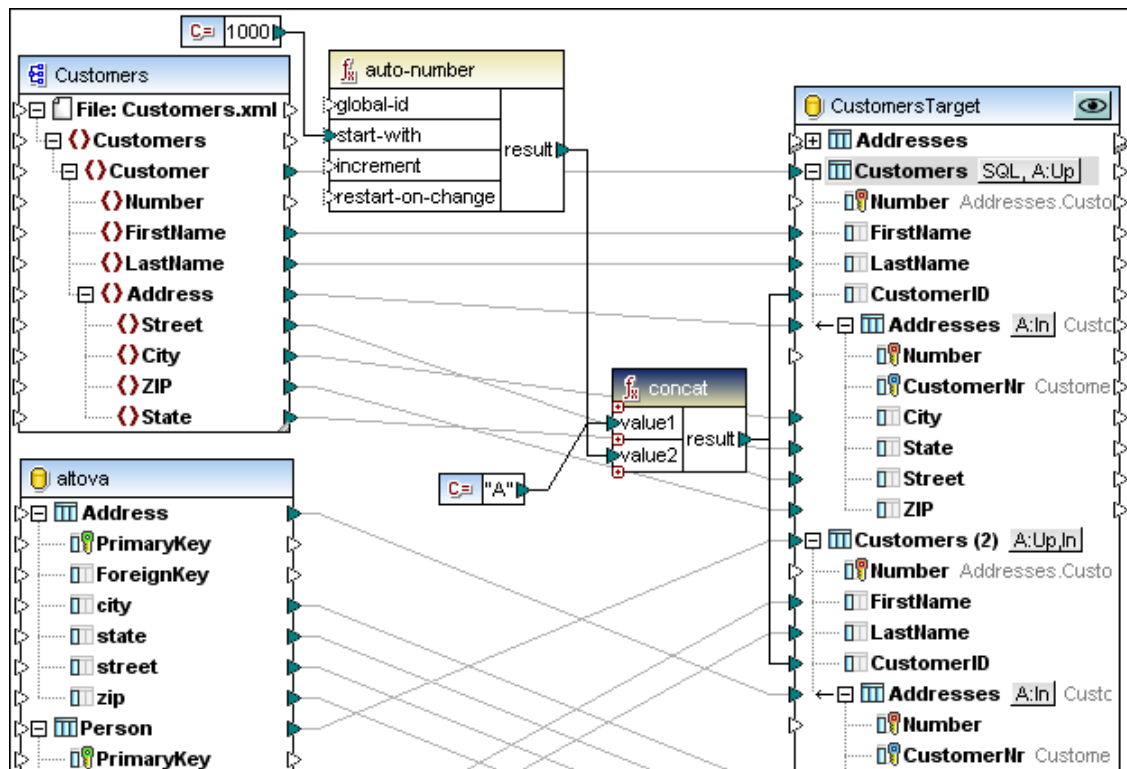
Alles einfügen...	Aktualisieren, wenn...	Löschen, wenn...	
Daten der Parent-Tabelle (Department-Datensätze) werden aktualisiert.			
Child-Daten jener Tabellen, die die Bedingung "Aktualisieren, wenn..." erfüllen, werden gelöscht (Person-Datensätze).	Wenn die Option "Daten in Child-Tabellen löschen" aktiv ist, werden die Child-Daten (Person-Datensätze) aus allen Department-Elementen gelöscht. Alle Personendatensätze für alle Abteilungen, die in der XML-Quelldatei eine PrimaryKey-Entsprechung haben, werden gelöscht, d.h. selbst Personendatensätze der Datenbank, die in der XML-Quelldatei keine Entsprechung haben, werden gelöscht.	Wenn die Option "Daten in Child-Tabellen löschen" nicht aktiv ist, werden die Child-Daten jener Tabellen, die die "Aktualisieren, wenn..." Bedingung erfüllen (Person-Datensätze) gelöscht.	
Bestehende Datenbanktabellen, die in der XML-Eingabedatei keine Entsprechung haben (keine Quellschlüssel für den Vergleich "Aktualisieren, wenn...") (Engineering in diesem Fall), werden beibehalten.			
Alle Person-Datensätze aus der XML-Eingabeinstanz - auch solche, die in der Datenbank noch nicht vorhanden sind, werden eingefügt.	Die für die Personentabelle definierte "Aktualisieren, wenn..."-Bedingung schlägt fehl, weil alle Personendatensätze in der Datenbank durch die Option "Daten in Child-Tabellen löschen" gelöscht wurden. Es gibt keine Möglichkeit, die Primärschlüssel der Datenbank mit denen der XML-Daten zu vergleichen, da die Datenbankschlüssel gelöscht wurden. Es werden keine Datensätze aktualisiert.	Die Daten der Child-Tabelle (Person-Datensätze) werden gelöscht, bevor die Tabellenaktion "Löschen, wenn..." ausgeführt wird. Es werden keine Datensätze gelöscht.	Datenbankdatensätze, für die es keinen entsprechenden Personenschlüssel gibt, werden beibehalten.

Ein weiteres Beispiel, bei dem duplizierte Datenelemente sowie Einfügen, Aktualisieren und Transaktionen verwendet werden, finden Sie im Ordner [...\\MapForceExamples](#) in der Beispieldatei **Customers_DB.mfd**. In dem Beispiel wird gezeigt, wie XML-Schemas und Datenbankquellen auf Zieldatenbanken gemappt werden.

Im Beispiel:

- XML-Schema auf Datenbank:
In der Zieldatenbank gibt es Kunden und Adressen. Diese Einträge werden mit den neuen Daten aus dem XML-Quellschema/ -dokument aktualisiert. Anhand der Elemente FirstName und LastName werden die richtigen Zeilen in der Datenbank gesucht.
- Datenbank auf Datenbank:
Aus der Datenbankquelle werden Adress- und Personendaten in die Datenbank übernommen. Die Zieltabelle (Customers) wird dupliziert.

Für jeden Datensatz wird eine neue CustomerID erzeugt, wobei der Anfangswert A1000 ist.



17.2.6 Datenbankaktion: Löschen

Mit Hilfe der Tabellenaktion "Löschen, wenn..." können Sie selektiv Daten aus Tabellen löschen. Dazu werden bestimmte Datenelemente/Felder der Quell- und der Zielkomponente für den Vergleich ausgewählt. Je nach Ergebnis des Vergleichs wird dann die jeweils definierte Tabellenaktion ausgeführt.

Bitte beachten Sie: Diese Tabellenaktion ist nicht zu verwechseln mit der im Dialogfeld der Tabellenaktion verfügbaren Option **Daten in Child-Tabellen löschen**. Die Tabellenaktion "Löschen, wenn..." wirkt sich nur auf die Tabelle aus, für die diese Aktion definiert wurde, andere Tabellen sind davon nicht betroffen.

Ziel

In diesem Abschnitt des Tutorials lernen Sie, wie man mit Hilfe von MapForce Daten aus Datenbanktabellen löscht. Insbesondere werden die folgenden Schritte beschrieben:

- Löschen bestehender Personendatensätze in der Datenbank und
- Einfügen neuer Personendatensätze aus der XML-Eingabedatei.
-

In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Datenbank einfügen: Dieser Befehl befindet sich in der Symbolleiste "Komponente einfügen" und im Menü **Einfügen**. Klicken Sie auf diesen Befehl, um ein Dialogfeld **Datenbank auswählen** zu öffnen, in dem Sie eine Datenbankverbindung definieren und auswählen können, welche Datenbanktabellen in das MapForce Mapping inkludiert werden sollen.



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identischer Sub-Einträge ein- bzw. auszuschalten.



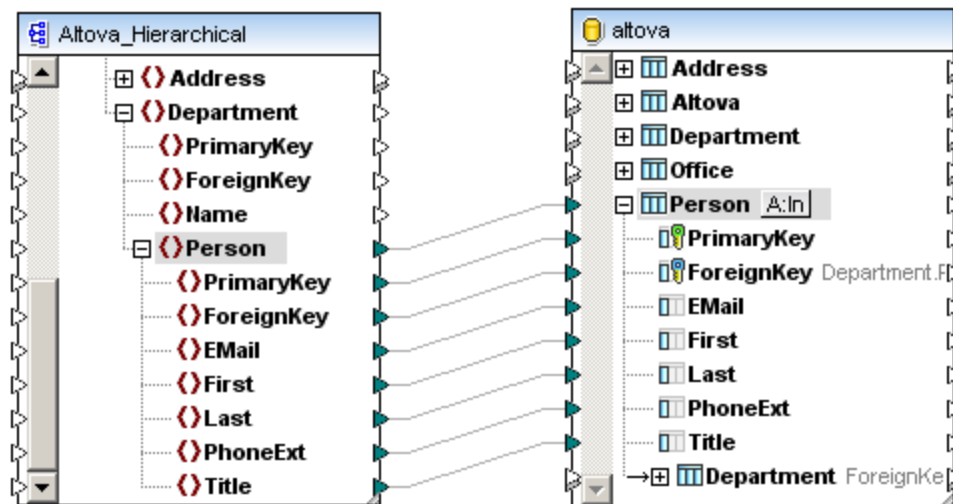
Datenbankaktionen: Dieser Befehl befindet sich im Menü **Komponente** einer Datenbankkomponente und wird in der Datenbankkomponente in Form einer Schaltfläche rechts vom Tabellennamen angezeigt. Klicken Sie auf diesen Befehl, um das Dialogfeld **Datenbankaktionen** aufzurufen, in dem Sie definieren können, welche Aktionen für die einzelnen Datensätze ausgeführt werden sollen.



SQL-Skript ausführen: Dieser Befehl befindet sich in der Symbolleiste "Ausgabevorschau" und im Menü "Ausgabe". Klicken Sie auf diesen Befehl, um das im Ausgabefenster generierte SQL-Skript auszuführen.

So richten Sie das Mapping ein und definieren die Datenbankaktionen für "Löschen, wenn...":

1. Fügen Sie das Schema `Altova_Hierarchical` ein (und weisen Sie die Datei **altova-cmpy-extra.xml** als XML-Eingabeinstanz zu).
2. Fügen Sie die MS Access-Datenbank **altova.mdb** in das Mapping ein und wählen Sie alle Benutzertabellen aus.



3. Wählen Sie die Menüoption **Verbindung | Idente Sub-Einträge automatisch verbinden**.
4. Klicken Sie in der XML-Quelldatei auf das Element Person und ziehen Sie den Konnektor zum Element Person der Datenbank. Stellen Sie sicher, dass Sie die Verbindung mit der "Root"-Tabelle **Person** herstellen.
Alle übereinstimmenden Child-Elemente werden automatisch gemappt.
5. Klicken Sie auf das **Person** Tabellenaktionssymbol **A:In**, um das Dialogfeld **Datenbankaktionen...** zu öffnen.
6. Klicken Sie auf die Auswahlliste "Aktion an Input-Daten" und wählen Sie **Löschen, wenn...** aus.
7. Klicken Sie auf die Schaltfläche **Aktion anhängen**.
Daraufhin wird automatisch eine neue Tabellenaktionsspalte mit der Tabellenaktion "Rest einfügen" eingefügt.

Datenbankaktionen - Person

Vor dem ersten Datensatz auszuführende SQL-Anweisung

☒ Keine
☐ Alle Datensätze LÖSCHEN ☐ auch alle Datensätze in Child-Tabellen löschen
☐ Benutzerdefinierte SQL:

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle D jeweilige Aktion ausgeführt.

Aktion an Input-Daten	Löschen, wenn...	
PrimaryKey	gleich	Rest einfügen gemappter Wert
ForeignKey		gemappter Wert
EMail		gemappter Wert

8. Klicken Sie auf die Auswahlliste "PrimaryKey" und wählen Sie die Option **gleich**.
Die Primärschlüssel von Quelle und Ziel werden mit Hilfe des "gleich"-Operators verglichen. Wenn die beiden Schlüssel identisch sind, werden die gemappten Felder der Personentabellen gelöscht. Im Anschluss daran wird die nächste Tabellenaktion gestartet, in diesem Fall "Rest einfügen".

Rest einfügen fügt all jene Datensätze aus der XML-Quelldatei ein, die keine Entsprechung (Schlüssel/Feld) in der Datenbank haben.

9. Klicken Sie auf OK, um das Dialogfeld zu schließen.
Das Tabellenaktionssymbol zeigt nun **A:De,In** an.



Anmerkung: Wenn in den Zeilen für PrimaryKey oder ForeignKey kein Auswahllistenabwärtspfeil zur Verfügung steht, müssen Sie die Eigenschaften zur Behandlung dieses Schlüssels ändern, siehe [Tabellenaktionen: Schlüsseleinstellungen](#).

10. Klicken Sie auf die Schaltfläche Ausgabe unten im Mapping-Fenster, um das bei diesem Mapping erzeugte SQL Script zu sehen. Das Script führt das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.
11. Klicken Sie in der Funktionsleiste auf das Symbol "SQL-Skript ausführen" , um das Script zu starten und die Datensätze der Datenbank zu aktualisieren.
12. Öffnen Sie die Altova-Datenbank in MS Access und doppelklicken Sie auf die Tabelle **Person**, um zu sehen, wie sich die Aktionen darauf ausgewirkt haben.

Person : Table					
	PrimaryKey	ForeignKey	E-Mail	First	Last
▶	6	3	f.landis@nanonull.com	Fred	Landis
	7	3	m.landis@nanonull.co	Michelle	Butler
	8	3	t.little@nanonull.com	Ted	Little
	9	3	a.way@nanonull.com	Ann	Way
	10	3	l.gardner@nanonull.cc	Liz	Gardner
	11	3	p.smith@nanonull.cor	Paul	Smith
	12	4	a.martin@nanonull.co	Alex	Martin
	13	4	g.hammer@nanonull.c	George	Hammer
	30	1	c.Cicada@microtech.i	Camilla	Cicada
	31	1	c.corrigan@microtech	Carol	Corrigan
*					

Verarbeitungsreihenfolge

Mit der Bedingung **Löschen, wenn...** wird überprüft, ob die Primärschlüssel der Datenelemente der Quell- und Zielkomponente identisch sind oder nicht.

Tabelle Person:

- Wenn die Bedingung **true ist**, werden alle Personendatensätze, deren Schlüssel identisch sind, gelöscht. Gibt es in der Datenbank Datensätze, für die es in der Quelldatei keinen identen Schlüssel/kein identes Feld gibt, werden diese Datensätze nicht gelöscht und bleiben unverändert.
- Wenn die Bedingung **false ist**, d.h. es gibt Quellschlüssel, die in der Zieldatenbank keine Entsprechung haben, dann geht MapForce zur nächsten Tabellenaktionsspalte weiter, nämlich: **Rest einfügen...** Mit dieser Aktion werden die neuen Personendatensätze, falls vorhanden, in die Personentabelle eingefügt. In diesem Beispiel werden zwei neuen Personendatensätze mit dem Primärschlüssel 30 bzw. 31 zur Abteilung Administration hinzugefügt.

Im Abschnitt "[Aktualisieren, wenn...-Kombinationen](#)" finden Sie zwei weitere Beispiele für die Tabellenaktion "Löschen, wenn...".

17.2.7 Datenbankaktion: Ignorieren

Mit Hilfe der Tabellenaktion "Ignorieren, wenn...", können bestimmte vom Mapping erzeugte Datensätze selektiv ignoriert werden. Dazu wird eine SELECT-Anweisung mit der definierten Bedingung erstellt. Wenn die identischen Daten von der Anweisung gefunden werden, wird der entsprechende gemappte Input-Datensatz ignoriert.

Mit Hilfe dieser Aktion zusammen mit der folgenden "Rest einfügen" Aktion werden alle Input-Datensätze, die bereits in der Datenbank vorhanden sind, ignoriert und neue werden eingefügt.

Ziel

In diesem Abschnitt des Tutorials erfahren Sie, wie sich bestimmte Datenbankdatensätze bei Mappings in MapForce ignorieren lassen. Insbesondere werden die folgenden Schritte beschrieben:

- Ignorieren doppelt vorhandener Person-Datensätze in der Datenbank und
- Einfügen neuer Person-Datensätze aus der XML-Input-Datei.

In diesem Abschnitt verwendete Befehle



XML Schema/Datei einfügen: Klicken Sie auf diese Schaltfläche, um das Windows-Standarddialogfeld "Öffnen" aufzurufen und die Datei aus Ihrem Dateisystem auszuwählen.



Datenbank einfügen: Dieser Befehl befindet sich in der Symbolleiste "Komponente einfügen" und im Menü **Einfügen**. Klicken Sie auf diesen Befehl, um ein Dialogfeld **Datenbank auswählen** zu öffnen, in dem Sie eine Datenbankverbindung definieren und auswählen können, welche Datenbanktabellen in das MapForce Mapping inkludiert werden sollen.



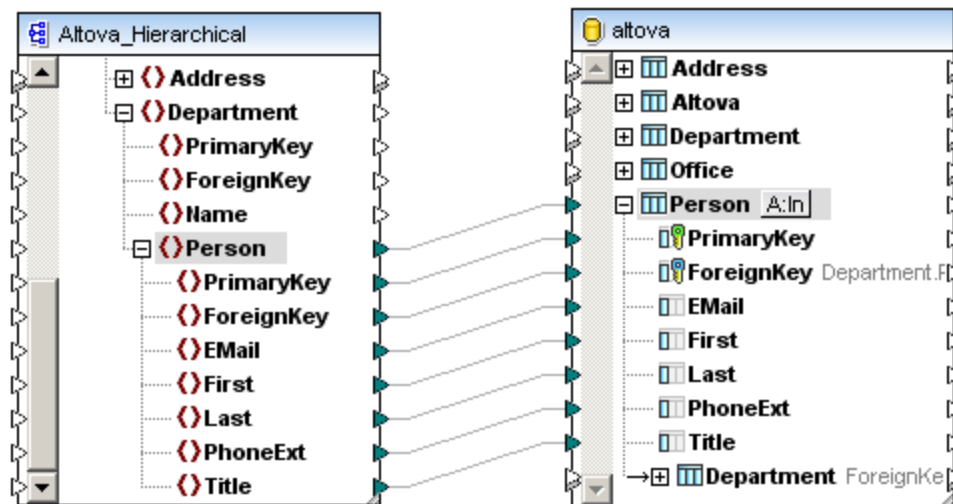
Datenbankaktionen: Dieser Befehl befindet sich im Menü **Komponente** einer Datenbankkomponente und wird in der Datenbankkomponente in Form einer Schaltfläche rechts vom Tabellennamen angezeigt. Klicken Sie auf diesen Befehl um das Dialogfeld **Datenbankaktionen** aufzurufen, in dem Sie definieren können, welche Aktionen für die einzelnen Datensätze ausgeführt werden sollen.



Identische Sub-Einträge automatisch verbinden: Klicken Sie auf diese Schaltfläche, um die automatische Verbindung identischer Sub-Einträge ein- bzw. auszuschalten.

So richten Sie das Mapping ein und definieren die Datenbankaktionen für "Ignorieren, wenn...":

1. Fügen Sie das Schema `Altova_Hierarchical` ein (und weisen Sie die Datei **altova-cmpy-extra.xml** als XML-Eingabeinstanz zu).
2. Fügen Sie die MS Access-Datenbank **altova.mdb** in das Mapping ein und wählen Sie alle Benutzertabellen aus.



3. Wählen Sie die Menüoption **Verbindung | Idente Sub-Einträge automatisch verbinden**.
4. Klicken Sie in der XML-Quelldatei auf das Element Person und ziehen Sie den Konnektor zum Element Person der Datenbank. Stellen Sie sicher, dass Sie die Verbindung mit der "Root"-Tabelle **Person** herstellen.
Alle übereinstimmenden Child-Elemente werden automatisch gemappt.
5. Klicken Sie auf das **Person** Tabellenaktionssymbol **A:In**, um das Dialogfeld **Datenbankaktionen...** zu öffnen.
6. Klicken Sie auf die Auswahlliste "Aktion an Input-Daten" und wählen Sie **Ignorieren, wenn...** aus.
7. Klicken Sie auf die Schaltfläche **Aktion anhängen**.
Daraufhin wird automatisch eine neue Tabellenaktionsspalte mit der Tabellenaktion "Rest einfügen" eingefügt

Datenbankaktionen - Person

Vor dem ersten Datensatz auszuführende SQL-Anweisung

☒ Keine
☐ Alle Datensätze LÖSCHEN ☐ auch alle Datensätze in Child-Tabellen löschen
☐ Benutzerdefinierte SQL:

Für jeden Datensatz auszuführende Aktionen

Alle Eingabedaten werden hier mit den Daten der Datenbank-Tabelle verglichen. Stimmen alle ausgeführt.

Aktion an Input-Daten	Ignorieren, wenn...	Rest einfügen
PrimaryKey		max() + 1
ForeignKey		gemappter Wert
EMail		gemappter Wert
First	gleich	gemappter Wert
Last	gleich	gemappter Wert
PhoneExt		gemappter Wert
Title		gemappter Wert

8. Klicken Sie auf die Auswahlliste "PrimaryKey" und wählen Sie die Option **gleich**.
9. Klicken Sie auf OK, um das Dialogfeld zu schließen.

Verarbeitungsreihenfolge

Die Tabellenaktion **Ignorieren, wenn...** vergleicht die First- und Last-Felder der XML-Quelldatei mit denselben Feldern in der Datenbank:

- Wenn die Daten beider Felder in der Datenbank gefunden werden, werden die **gemappten Daten ignoriert** und die Tabellenaktion "Rest einfügen" wird nicht verarbeitet.
- Wenn der Vergleich bei den Feldern "First" oder "Last" fehlschlägt, wird mit Hilfe des PrimaryKey-Eintrags "max()+ 1" ein neuer Datensatz in der Datenbank generiert.

17.2.8 Generieren von Datenbankausgabewerten

Mit Hilfe der Generierungsfunktionen für die Programmiersprachen sowie mit Hilfe des Built-In-Ausführungsprozessors können Werte für Datenbankfelder generiert werden, für die es keine Eingabedaten aus der Schema-, Datenbank- oder EDI-Quellkomponente gibt.

Mit "auto-number" und "create-guid" können Werte für Felder generiert werden.

[auto-number](#) (core | generator functions) - steht auch für den Built-In-Ausführungsprozessor zur Verfügung
wird normalerweise zum Generieren von Primärschlüsselwerten für ein numerisches Feld verwendet.

[create-guid](#) (lang | generator functions)
erzeugt einen global eindeutigen Bezeichner (als hex-kodierten String) für ein bestimmtes Feld.

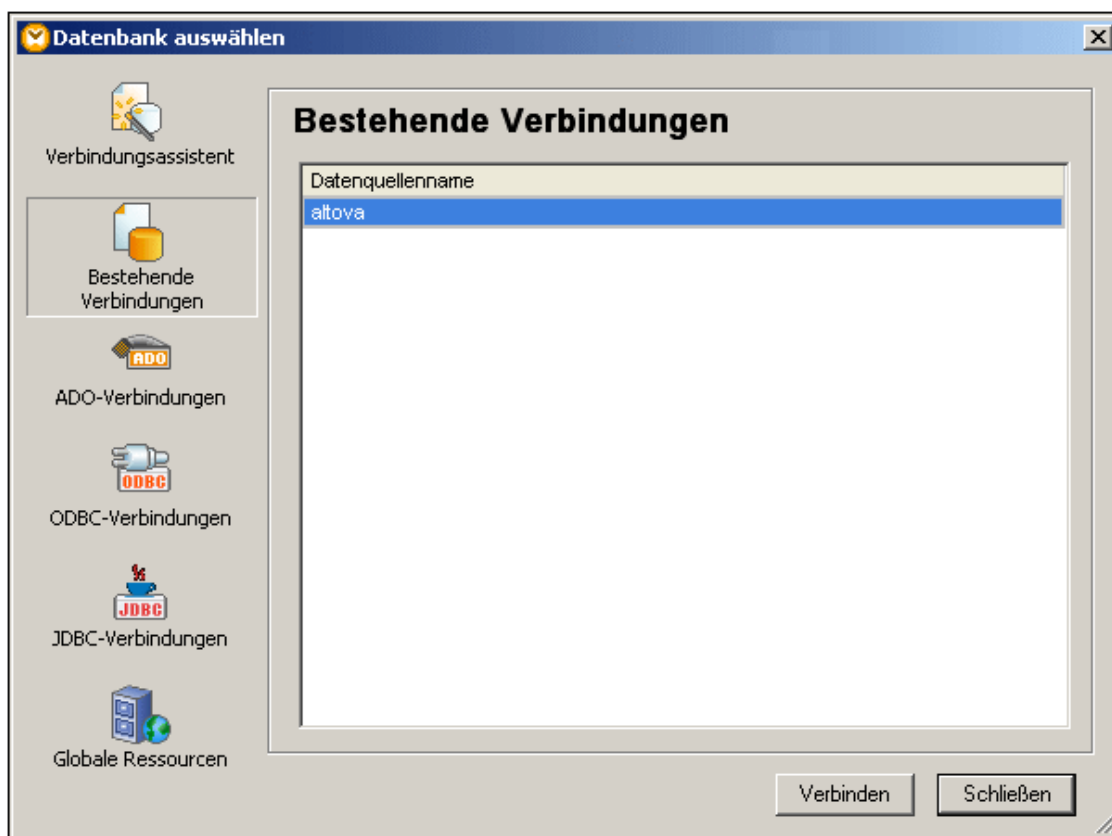
17.3 Verwendung des Verbindungsassistenten

Der Verbindungsassistent hilft Ihnen beim Herstellen einer Verbindung zu einer Datenbank und stellt eine Reihe von Verbindungsarten zur Auswahl. Sie können damit schnell eine Verbindung zu einem beliebigen Datenbanktyp in der Liste herstellen.



Im Fenster **Bestehende Verbindungen** werden alle derzeit **aktiven** Datenbankverbindungen aufgelistet.

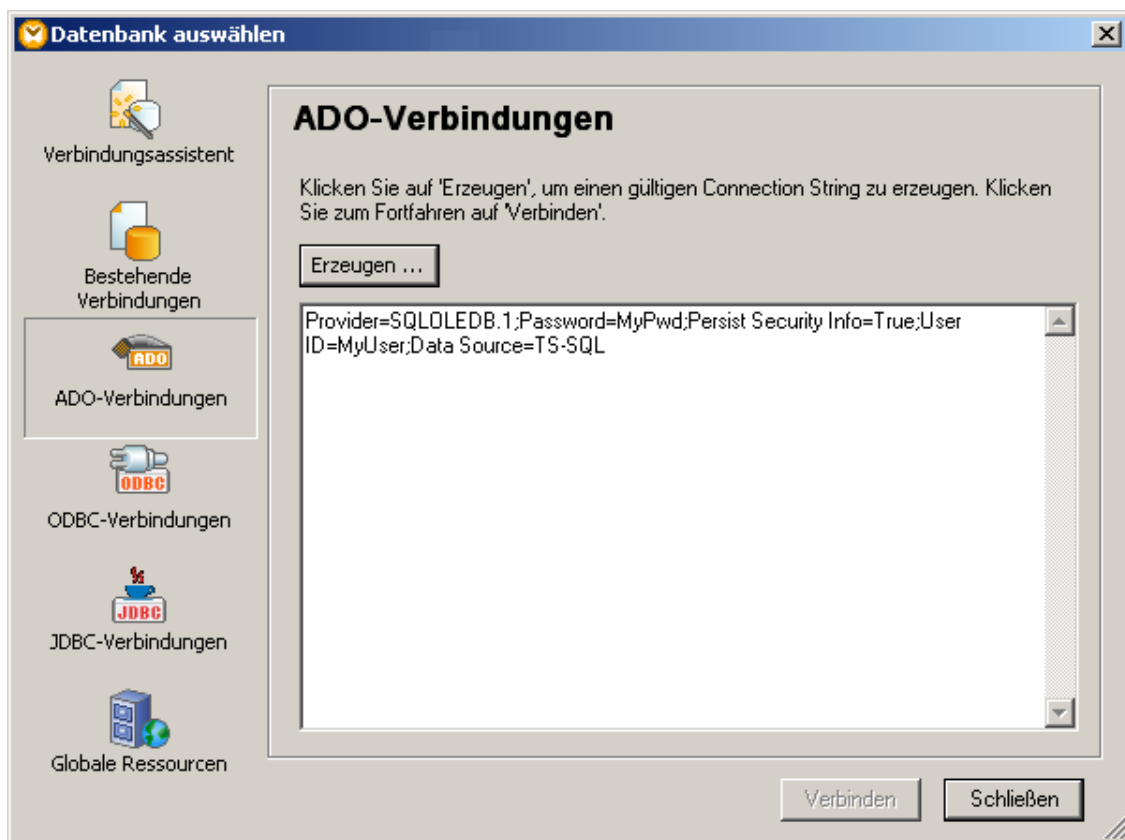
- Wählen Sie eine Verbindung in der Liste aus und klicken Sie auf **Verbinden**, um die Verbindung zu dieser Datenbank herzustellen.



Im Fenster **ADO-Verbindungen** können Sie mittels der Schaltflächen **Erzeugen** eine ADO-Verbindung zu einer Datenbank herstellen.

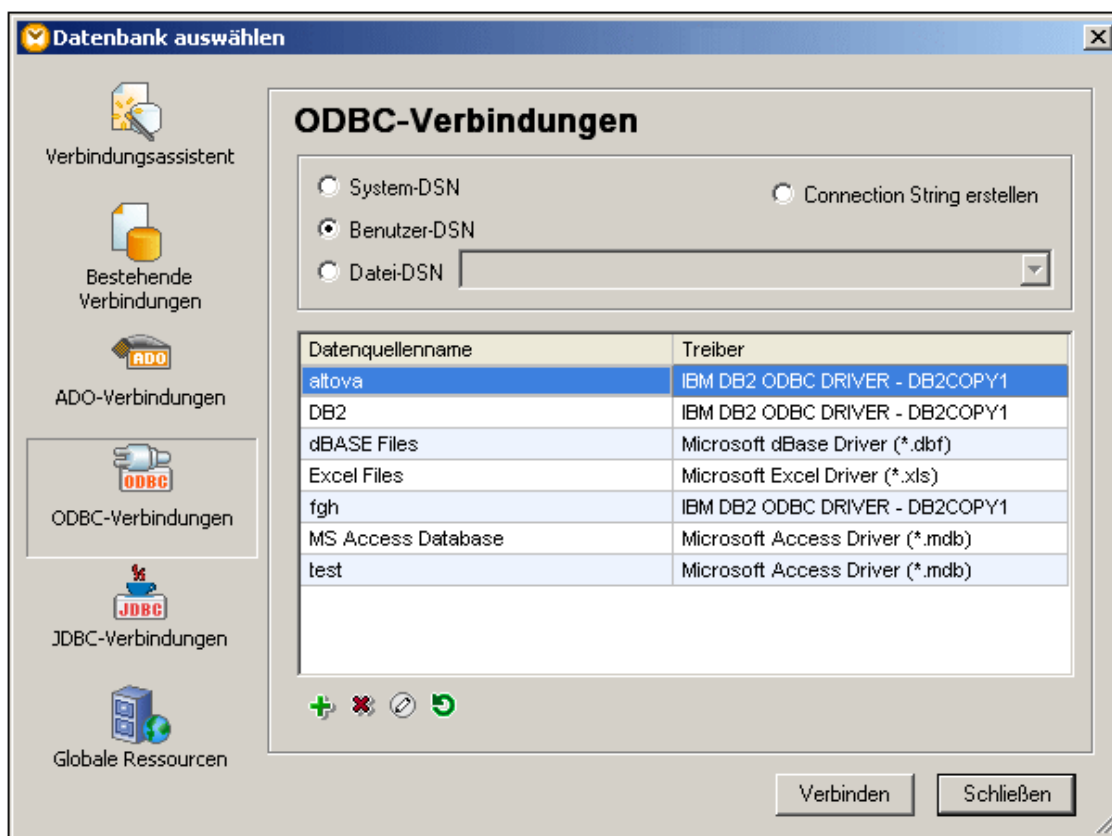
- Klicken Sie auf die Schaltfläche **Erzeugen** um den Connection String zu erstellen, welcher im Fenster angezeigt wird, sobald er definiert wurde und klicken Sie auf die Schaltfläche **Verbinden**, um eine Verbindung zur Datenbank herzustellen.

Eine ADO (ActiveX Data Objects)-Verbindung ohne Durchführung vorbereitender Schritte wie etwa die Erstellung eines DSN hergestellt werden. Verwenden Sie für MS Access-Datenbanken immer einer ADO-Verbindung, da ODBC Beziehungen nicht unterstützt.



Im Fenster **ODBC-Verbindung** können Sie eine ODBC (Open Database Connectivity)-Verbindung zu einer Datenbank herstellen. Sie können aus den folgenden Typen auswählen:

- System-DSN (Data Source Name): Diese Art von DSN kann von jedem verwendet werden, der Zugriff auf den Computer hat. DSN-Informationen sind in der Registrierdatei gespeichert.
- Benutzer-DSN: Diese Art von DSN wird für einen bestimmten Benutzer erstellt und ebenfalls in der Registrierdatei gespeichert.
- Datei-DSN: Bei dieser Art von DSN werden die DSN-Informationen in einer Textdatei mit DSN-Erweiterung gespeichert.



Bitte beachten Sie: Um eine ODBC-Verbindung zu erstellen, müssen Sie zuerst einen DSN erstellen. Das Listenfeld "Datenquelle" enthält alle zuvor erstellten DSNs.

Wenn Sie das Optionsfeld System-DSN oder Benutzer-DSN (Data Source Name) aktivieren, steht Ihnen im unteren Fensterbereich eine Reihe von Schaltflächen zur Erstellung neuer oder zum Aktualisieren bestehender DSNs zur Verfügung.



Neuen DSN erstellen: Erstellt einen neuen DSN.



Datenquellennamen bearbeiten: Dient zum Ändern der Einstellungen des in der Datenquellenliste oberhalb ausgewählten DSN.

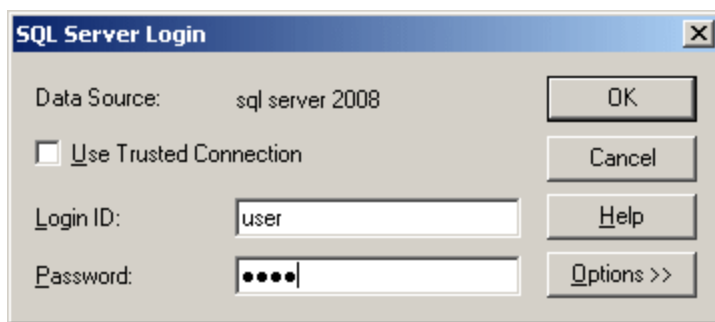


Datenquellennamen entfernen: Entfernt den gerade ausgewählten DSN aus der Liste der Datenquellen.

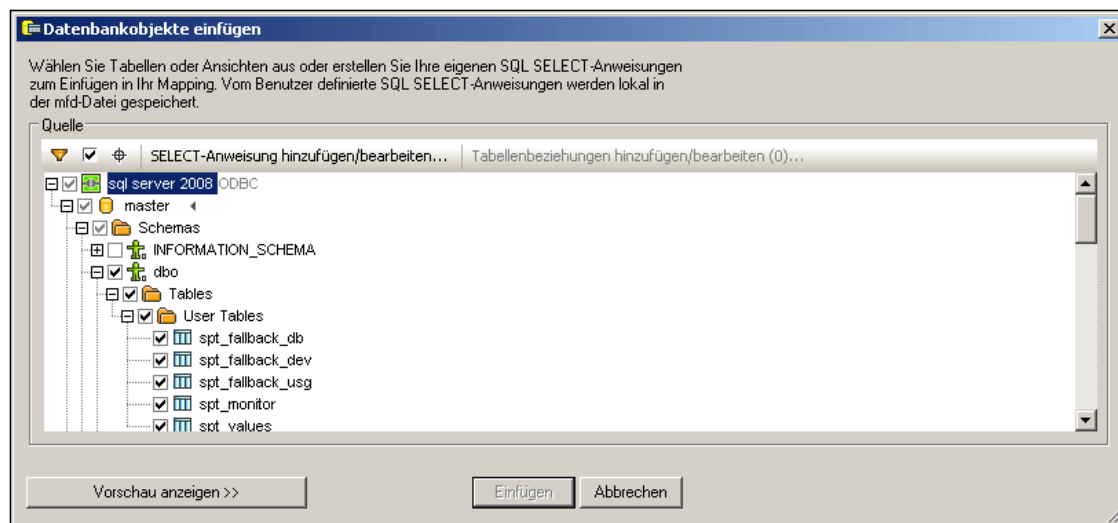


Datenquellennamen aktualisieren: Aktualisiert die Liste der Datenquellennamen.

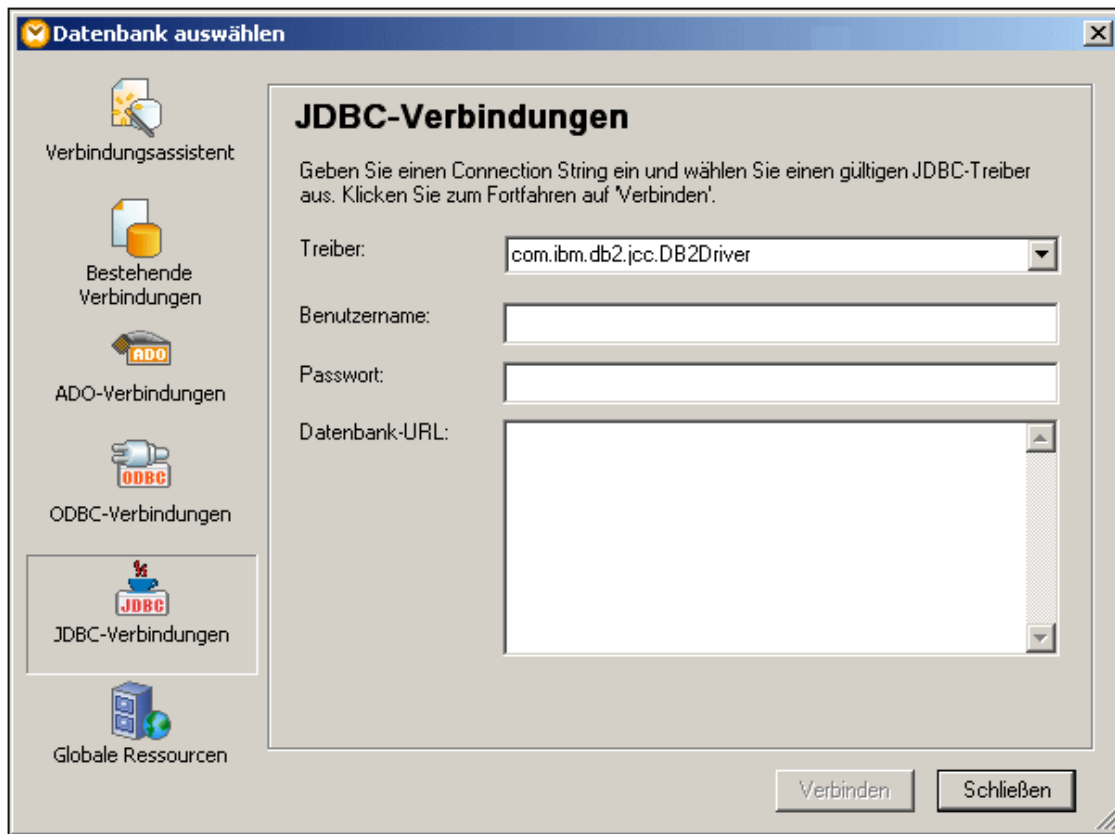
Wenn Sie im Listenfeld auf einen der Datenquellennamen klicken, wird das Dialogfeld für das entsprechende Datenbank-Login geöffnet.



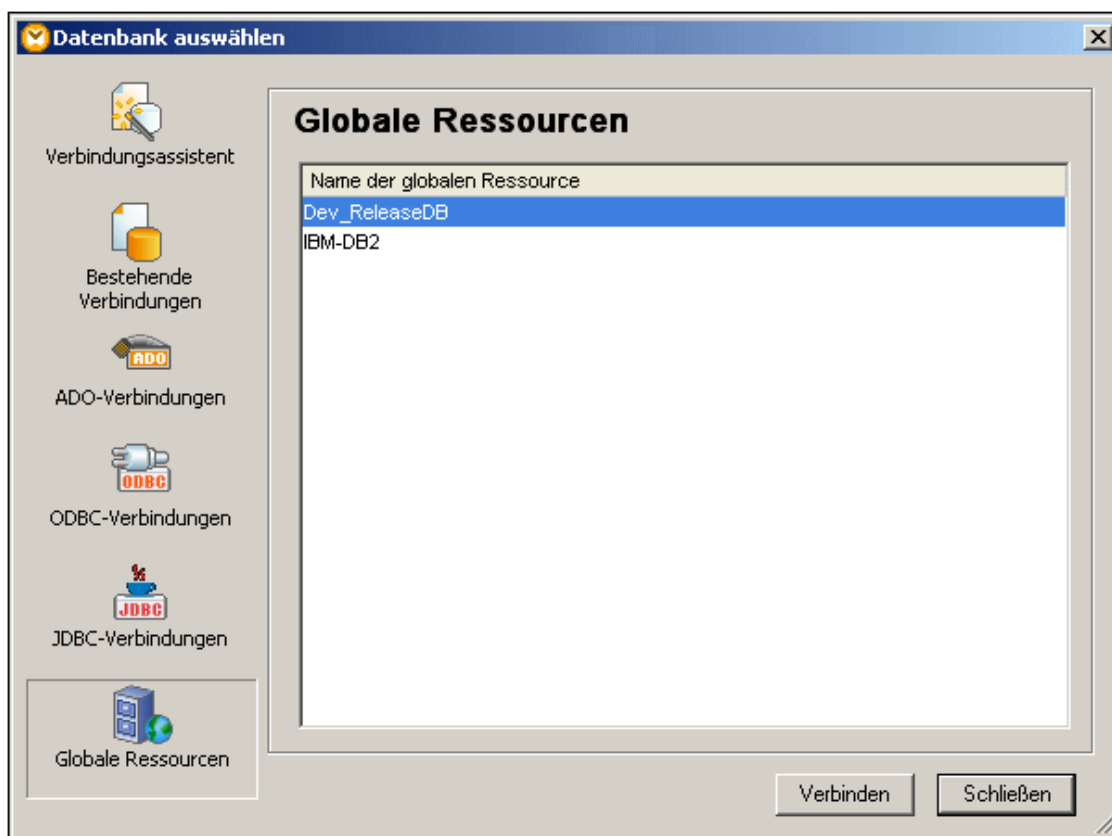
Geben Sie die benötigten Login-Daten ein und klicken Sie auf OK.



Im Fenster **JDBC-Verbindungen** können Sie über den JDBC-Treiber eine Verbindung zu verschiedenen Datenbanken herstellen. Nähere Informationen dazu finden Sie unter [JDBC-Verbindungen](#).



Im Bereich **Globale Ressourcen** können Sie einen zuvor definierten Datenbankalias für eine globale Ressource auswählen. Nähere Informationen dazu finden Sie unter: [Globale Ressourcen - Datenbanken](#).



17.4 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:

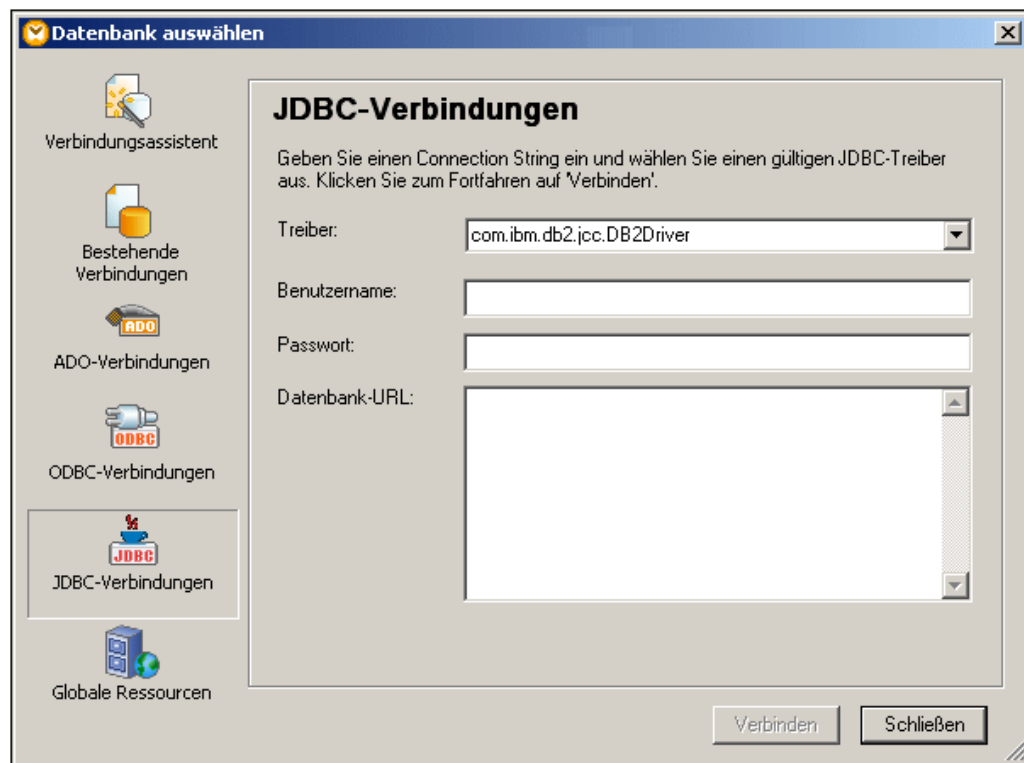
1. Vergewissern Sie sich, dass das Java Runtime Environment (JRE) installiert ist. Ist dies nicht der Fall, dann installieren Sie es. Verwenden Sie für einen 32-Bit-Rechner ein 32-Bit JRE und für einen 64-Bit-Rechner ein 64-Bit JRE.
2. Installieren Sie einen JDBC-Treiber. Dazu ist keine spezielle Installation 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). Bearbeiten Sie den Connection String 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 und es wird ein Dialogfeld angezeigt, in dem Sie die erforderlichen Datenbankinformationen auswählen können. Informationen darüber finden Sie im Abschnitt [Verwendung des Verbindungsassistenten](#).

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 MapForce 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 MapForce 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 MapForce 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.

17.5 Tabellenaktionen, Schlüsseinstellungen, Transaktionsverarbeitung

Im Dialogfeld [Datenbankaktionen](#) können Sie (d) eine optionale SQL-Anweisung, die vor jeder anderen Tabellenaktion auszuführen ist definieren, (ii) festlegen, welche Tabellenaktionen für jeden Datensatz, der vom Mapping in die Zieltabelle übertragen wird, auszuführen sind und (iii) die Datenbankschlüsseinstellungen zum Einfügen neuer Datensätze definieren.

Vor dem ersten Datensatz auszuführende SQL-Anweisung

Im Bereich **Vor dem ersten Datensatz auszuführende SQL-Anweisung** können Sie SQL-Anweisungen definieren, die ausgeführt werden sollen, bevor der erste Datensatz aus der Datenbank abgerufen wird. Aktivieren Sie das gewünschte Optionsfeld:

- **Keine** (Standardeinstellung): Es wird keine Aktion ausgeführt
- **Alle Datensätze LÖSCHEN**: Sie können alle Datensätze der Tabelle löschen, bevor eine bestimmte unter den Aktionen für die einzelnen Datensatzgruppen definierte Tabellenaktion ausgeführt wird. Aktivieren Sie das Kontrollkästchen "Daten in Child-Tabellen löschen", wenn auch die Daten in den Child-Tabellen der ausgewählten Tabelle gelöscht werden sollen.
- **Benutzerdefinierte SQL**: Sie können eine benutzerdefinierte SQL-Anweisung schreiben, die sich auf die gesamte Tabelle auswirkt.

Für die einzelnen Datensätze auszuführende Aktionen

Die Tabellenaktionen definieren, wie die gemappten Datensätze verarbeitet werden. Es stehen mehrere Optionen zur Verfügung: [Einfügen](#), [Aktualisieren](#), [Löschen](#) und [Ignorieren](#) werden über

die Einträge der Auswahllisten zu "Aktion an Input-Daten" ausgewählt.

Nähere Informationen dazu finden Sie unter [Tabellenaktionen](#).

17.5.1 Tabellenaktionen

Im Bereich "Für jeden Datensatz auszuführende Aktionen" können Sie die Tabellenaktionen definieren, mit denen bestimmte gemappte Datensätze bearbeitet werden sollen. MapForce unterstützt die Tabellenaktionen [Einfügen](#), [Aktualisieren](#), [Löschen](#) und [Ignorieren](#). Für die Tabellenaktionen "Aktualisieren", "Löschen" und "Ignorieren" wird eine **Bedingung** benötigt, die definiert, auf welche Datensätze sie angewendet werden sollen. Die Quell- und Zieldaten werden anhand eines oder mehrerer Felder verglichen, um festzustellen, ob die Tabellenaktion ausgeführt werden soll. Die Aktion "Einfügen" hat keine Bedingung aber **Schlüsseleinstellungen**, mit denen definiert wird, wie der Primärschlüssel generiert wird.

Im Dialogfeld "Tabelleaktionen" werden folgende Einstellungen für jede gemappte Datenbankspalte definiert:

- Felder, die verglichen werden sollen (z.B. PrimaryKey, FirstName usw.)
- für den Vergleich zu verwendenden Operatoren (equal, equal ignoring case) und
- die auszuführende Aktion, wenn alle Bedingungen der einzelnen Spalten erfüllt werden

Tabelleaktionen werden von links nach rechts verarbeitet. Im oben stehenden Beispiel wird zuerst die **Aktualisieren, wenn...**-Spalte verarbeitet. Wenn die Aktualisierungsbedingung nicht erfüllt wird, wird die folgende Aktion verarbeitet, z.B. die Spalte **Rest einfügen**.

- Damit die Tabelleaktion ausgeführt wird, müssen **alle** definierten Bedingungen einer Spalte erfüllt werden. Ist dies der Fall, werden alle **gemappten** Felder aktualisiert, d.h. alle Elemente zwischen denen zwischen Quelle und Ziel im Mapping-Fenster ein Konnektor vorhanden ist, werden aktualisiert. Alle nachfolgenden Tabelleaktionen (die sich rechts von einer Aktion befinden, deren Bedingung erfüllt wurde), werden für diesen Datensatz ignoriert.

- Wenn eine Bedingung nicht erfüllt wird, wird die Tabellenaktion für diese Spalte übersprungen und die nächste Spalte wird verarbeitet.
- Ist keine der Bedingungen "true", wird keine Tabellenaktion ausgeführt.

Bitte beachten Sie:

Alle nach **Alles einfügen/Rest einfügen** definierten Tabellenaktionen werden nie ausgeführt, da es keine Spaltenbedingungen für Einfügeaktionen gibt. In diesem Fall erscheint ein Dialogfeld, in dem Sie darauf hingewiesen werden, dass die nachfolgenden Tabellenaktionsspalten gelöscht werden.

Daten in Child-Tabellen löschen:

- Standardeinstellung bei Auswahl der **Aktualisieren, wenn...**-Aktion.
- Ist notwendig, wenn sich die Anzahl der Datensätze in der Quelldatei von denen der Zieldatenbank unterscheidet.
- Dient dazu, die Datenbank zu synchronisieren (keine allein stehenden Daten in den Child-Tabellen)

Auswirkung:

- Die Aktualisieren, wenn...-Bedingung wird erfüllt, wenn in der XML-Quelldatei ein übereinstimmender Schlüssel (oder ein anderes identes Feld) vorhanden ist. Alle **Child-Daten** der Parent-Tabelle werden gelöscht.
- Aktualisieren, wenn...wählt die Parent-Tabelle und somit auch die damit verknüpften Child-Tabellen aus, auf die sich die Option "Daten in Child-Tabellen löschen" auswirkt.
- Wenn die Aktualisieren-Bedingung (in der Parent-Tabelle) nicht erfüllt wird, d.h. wenn es in der XML-Quelldatei keinen übereinstimmenden Schlüssel/kein identes Feld gibt, werden die Child-Daten nicht gelöscht.
- Bestehende Datensätze, die in der Quelldatei keine Entsprechung haben, werden nicht aus der Datenbank gelöscht, sondern beibehalten.

Input-Child-Daten ignorieren:

Verwenden Sie diese Option, um bestimmte Tabellendaten zu aktualisieren, ohne dass Child-Tabellen/Child-Datensätze dieser Tabelle dadurch betroffen sind.

So könnte Ihr Mapping z.B. aus 3 Quelldatensätzen und 2 Datensätzen der Zieldatenbank bestehen.

Sie müssten daher:

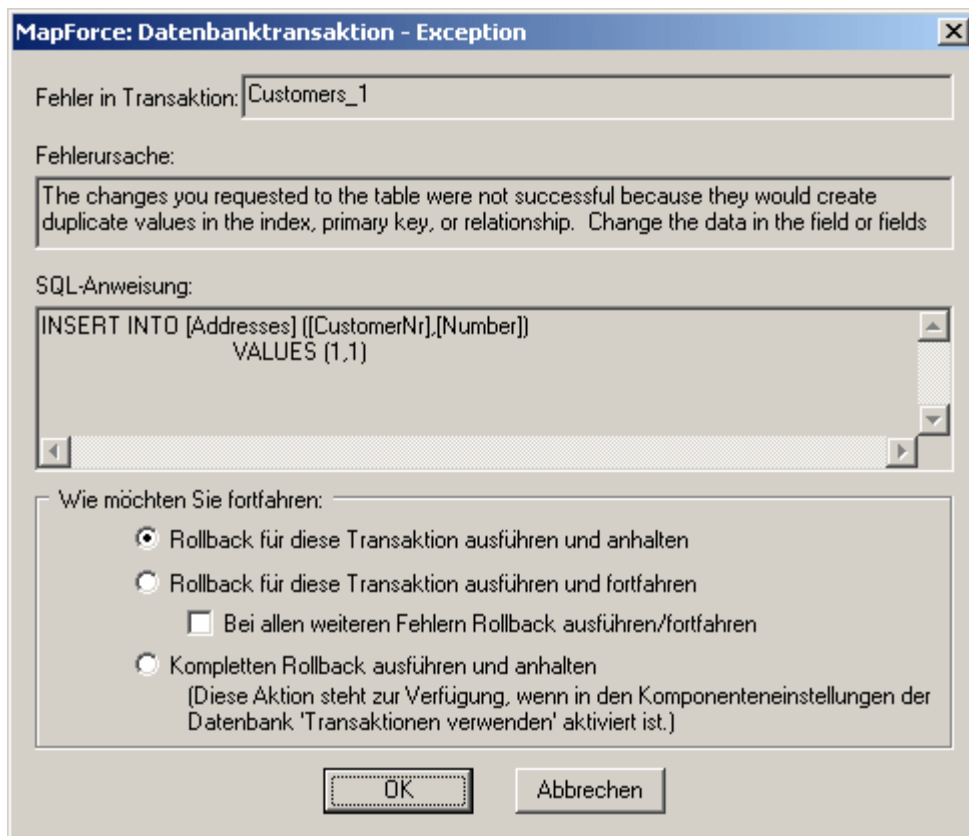
- eine **Aktualisieren, wenn...**-Bedingung definieren, um die bestehenden Datensätze zu aktualisieren
- das Kontrollkästchen **Input-Child-Daten ignorieren** der **Aktualisieren, wenn...-Spalte** aktivieren, damit damit verknüpfte Child-Datensätze ignoriert werden und
- für neue, einzufügende Datensätze eine **Rest einfügen...**-Bedingung definieren.

Transaktionen verwenden:

Über das Kontrollkästchen "Transaktionen verwenden" können Sie definieren, was geschehen soll, wenn eine Datenbankaktion aus irgendeinem Grund nicht durchgeführt werden kann. Im Falle einer solchen Ausnahme wird ein Dialogfeld geöffnet, in dem Sie aufgefordert werden, die weitere Vorgehensweise anzugeben. Wählen Sie dann die gewünschte Option aus und klicken Sie auf OK, um fortzufahren. Wenn Sie diese Option (über das Dialogfeld "Datenbankaktionen") für eine bestimmte Tabelle aktivieren, so erfolgt im Falle eines Fehlers ein Rollback für diese spezifische Datenbanktabelle.

Sie können die Transaktionseinstellung auch für die Datenbankkomponente aktivieren, indem

Sie im Dialogfeld Komponenteneinstellungen der entsprechenden Datenbankkomponente mit der rechten Maustaste darauf klicken. In diesem Fall wird der Rollback für alle Tabellen aktiviert.



Es sind **keine Transaktionsoptionen** eingestellt:

Wenn das Kontrollkästchen "Transaktionen verwenden" in den Tabellenoptionen bzw. in den Komponenteneinstellungen nicht aktiviert wurde, und ein Fehler auftritt, geschieht Folgendes:

- Der Vorgang bricht an dem Punkt ab, an dem der Fehler auftritt. Alle bisher erfolgreichen SQL-Anweisungen werden ausgeführt und die Ergebnisse werden in der Datenbank gespeichert.

Die Transaktionsoption ist auf Ebene der **Datenbankkomponente** aktiviert:

- Der Vorgang bricht an dem Punkt ab, an dem der Fehler auftritt. Alle zuvor erfolgreich ausgeführten SQL-Anweisungen werden rückgängig gemacht. In der Datenbank werden keine Änderungen vorgenommen. Alle zuvor erfolgreich ausgeführten SQL-Anweisungen für die **Datenbank** und alle ihre Tabellen können rückgängig gemacht werden.

Die Transaktionsoption ist auf Ebene der **Tabellenaktionen** aktiviert:

- Das Dialogfeld "MAPFORCE: Datenbanktransaktion - Ausnahmebedingung" wird angezeigt. Die Option "Kompletten Rollback ausführen und abbrechen" ist **deaktiviert**. Die fehlgeschlagene SQL-Anweisung kann für die jeweilige Tabelle rückgängig gemacht werden.

Die Transaktionsoption ist auf Ebene der **Datenbankkomponente** und auf Ebene der **Tabellenaktion** aktiviert:

- Das Dialogfeld "MAPFORCE: Datenbanktransaktion - Ausnahmebedingung" wird angezeigt. Die Option "Kompletten Rollback ausführen und abbrechen" ist **aktiviert**. Alle zuvor erfolgreich ausgeführten SQL-Anweisungen für die **Datenbank** und alle ihre Tabellen können rückgängig gemacht werden.

Wenn Sie auf die Schaltfläche **Abbrechen** klicken, wird die gesamte SQL-Anweisung rückgängig gemacht und abgebrochen.

Bitte beachten Sie:

Die Eingabeaufforderungen bei den Transaktionen werden nur dann angezeigt, wenn die Transaktion **interaktiv** ausgeführt wird!

Der **generierte Code** führt ein Rollback aus (und stoppt die Aktion), wenn der erste Fehler auftritt.

17.6 Datenbankfunktionsmatrix

Die folgenden Tabellen enthalten Informationen zu den Mapping-Funktionen, die MapForce für die gebräuchlichsten Datenbankformate unterstützt.

Die Tabellen enthalten die folgenden Informationen:

- Allgemeine Informationen über die Datenbank, wie Dienste und Authentifizierungsfragen
- Unterstützte Verbindungstypen
- SQL-Unterstützung von: Schemas, Join-Anweisungen usw.
- Unterstützte Transaktionsmethoden

17.6.1 Datenbank-Info - MS Access

	MS Access	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	n	implementiert im OLEDB-Provider oder ODBC-Treiber
	own authentication	j	Authentifizierung ist möglich
	Trusted authentication	n	
Verbindung:			
	OLE DB	j	
	OLE DB Connection-string issues	keine	
	ODBC	j	
	ODBC Connection-string issues	DBQ	muss angewendet werden
	ODBC Connection-string issues	DATABASE	darf nicht angewendet werden
	JDBC	j	via ODBC
	JDBC URL- issues	Keine	
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification	[] oder ""	
	support for DB-schemas	n	wird nicht von Access unterstützt
	identity support	j	
	MF: read back identity value	@@IDENTITY	
	sub select support	j	
	JOIN support	n	DB weist hier Einschränkungen auf
	MF: upper function	Ucase(..)	

SQL-Ausführung:			
	exec. multiple-stat. in one	n	
	command separator	--	
	special error handling	n	
	retrieve parameter types	?	
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	n	
	Nested transactions supported	n	wird nicht von Access unterstützt
	set transaction isolation	n	wird nicht von Access unterstützt
	MF: begin transaction	API-call	

	MF: commit transaction	API-call	
	MF: rollback transaction	API-call	
	MF: set save point	--	
	MF: rollback to save point	--	
	MF used init. Statements	Keine	

Der Datentyp "datetime" hat für XML-Schema und Access eine unterschiedliche semantische Bedeutung. In XML-Schema ist das Datum erforderlich, in Access ist es optional. Da MapForce intern XML-Schema-Datentypen verwendet, kann der Benutzer xs:time-Felder nicht direkt auf ein datetime-Feld in Access mappen.

Umgehungslösung:

Verwenden Sie die Funktion "datetime-from-date-and-time" und das Datum "1899-12-30".

Dies ist das Datum, das Access intern für reine Uhrzeitwerte verwendet.

17.6.2 Datenbank-Info - MS SQL Server

	MS SQLServer	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	j	
	own authentication	j	
	Trusted authentication	j	
Verbindung:			
	OLE DB	j	
	ODBC	j	
	ODBC connection-string issues	Auswahlmethode=Cursor	muss angewendet werden
	JDBC	j	
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification	[] oder ""	
	support for DB-schemas	j	
	identity support	j	
	MF: read back identity value	@@IDENTITY	
	sub select support	j	
	JOIN support	j	
	MF: upper function	UPPER()	
SQL-Ausführung :			
	exec. multiple-stat. in one	j	
	command separator	';' oder 'GO'	
	special error handling	n	
	retrieve parameter types	j	mit Einschränkungen
	special issues when using ?	j	DATETIME-Datentyp wird bei Verwendung von ODBC nicht unterstützt
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	j	Ein MUSS bei Verwendung von verschachtelten Transaktionen. Gemischte Verwendung von API-Transaktionsbehandlung und SQL-Transaktionsbefehlen ist nicht möglich
	Nested transactions supported	j	via SAVEPOINTS
	set transaction isolation	j	
	MF: begin transaction	BEGIN TRANSACTION	

	MF: commit transaction	COMMIT TRANSACTION	
	MF: rollback transaction	ROLLBACK TRANSACTION	
	MF: set save point	SAVEPOINT	
	MF: rollback to save point	ROLLBACK TO	
	MF used init. Statements	Keine	

17.6.3 Datenbank-Info - Oracle

	Oracle	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	j	
	own authentication	j	
	Trusted authentication	n	
Verbindung:			
	OLE DB	n	nicht offiziell von MapForce unterstützt
	ODBC	j	
	ODBC connection-string issues	DATABASE	Darf nicht angewendet werden
	JDBC	j	
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification	""	
	support for DB-schemas	j	
	identity support	n	muss Trigger verwenden
	MF: read back identity value	nicht unterstützt	
	sub select support	j	
	JOIN support	j	
	MF: upper function	UPPER()	
SQL-Ausführung:			
	exec. multiple-stat. in one	n	
	command separator	--	
	special error handling	n	
	retrieve parameter types		
	special issues when using ?	n	
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	n	
	Nested transactions supported	j	via SAVEPOINTS
	set transaction isolation	j	
	MF: begin transaction	API-call	
	MF: commit transaction	API-call	
	MF: rollback transaction	API-call	
	MF: set save point	SAVEPOINT	
	MF: rollback to save point	ROLLBACK TO SAVEPOINT	

	MF used init. Statements	Keine	

17.6.4 Datenbank-Info - MySQL

	MySQL	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	j	
	own authentication	j	
	Trusted authentication	n	
	special issues	TYPE=INNODB	für Tabellen bei Verwendung von Beziehungen, Transaktionen,...
Verbindung:			
	OLE DB	n	von MapForce nicht offiziell unterstützt
	ODBC	j	
	JDBC	j	via ODBC
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification	--	
	support for DB-schemas	j	
	identity support	j	
	MF: read back identity value	@@IDENTITY	
	sub select support	n	spezielle Implementierung für DELETE erforderlich
	JOIN support	j	
	MF: upper function	UPPER()	

SQL-Ausführung:			
	exec. multiple-stat. in one	n	
	command separator	--	
	special error handling	n	
	retrieve parameter types	y	mit Einschränkungen
	special issues when using ?	n	
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	n	
	Nested transactions supported	n	MySQL erzeugt keinen Fehler und fährt fort, wenn keine verschachtelten Transaktionen vorhanden sind
	set transaction isolation	j	
	MF: begin transaction	API-call	
	MF: commit transaction	API-call	
	MF: rollback transaction	API-call	

	MF: set save point	SAVEPOINT	
	MF: rollback to save point	ROLLBACK TO	
	MF used init. Statements	SET AUTOCOMMIT=0	

17.6.5 Datenbank-Info - Sybase

	Sybase	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	j	
	own authentication	j	
	Trusted authentication	n	
Verbindung:			
	OLE DB	n	von MapForce nicht offiziell unterstützt
	ODBC	j	
	ODBC connection-string issues	Auswahlmethode=Cursor	muss angewendet werden
	JDBC	j	via ODBC
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification		
	support for DB-schemas	j	
	identity support	j	
	MF: read back identity value	@@IDENTITY	
	sub select support	j	
	JOIN support	j	
	MF: upper function	UPPER()	

SQL-Ausführung:			
	exec. multiple-stat. in one	j	
	command separator	Keine	
	special error handling	n	
	retrieve parameter types		
	special issues when using ?	j	Bei Verwendung von ODBC sind nur ASCII-127 Zeichen in String-Konstanten zulässig
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	von MAPFORCE nicht unterstützt	
	Nested transactions supported	n	MySQL erzeugt keine Fehlermeldung und fährt fort, wenn keine verschachtelten Transaktionen vorhanden sind
	set transaction isolation	j	

	MF: begin transaction	API-call	
	MF: commit transaction	API-call	
	MF: rollback transaction	API-call	
	MF: set save point	SAVE TRANSACTION	
	MF: rollback to save point	ROLLBACK	
	MF used init. Statements	Keine	

Wenn Sie mit Hilfe der Sybase-Prozeduren 'sp_primarykey' und 'sp_foreignkey' Beziehungen zwischen Tabellen definiert haben, müssen Sie außerdem den Befehl ALTER TABLE verwenden, um einen Constraint zu der Tabelle hinzuzufügen, der die Sekundärschlüsselbeziehung beschreibt, damit die Primär-/Sekundärschlüsselbeziehung in MapForce aufscheint.

17.6.6 Datenbank-Info - IBM DB2

	IBM DB2	Unterstützung	Anmerkungen
Allgemein:			
	DB engine as service	j	
	own authentication	j	Verwendet lokale Windows-Benutzerkonten
	Trusted authentication	j	siehe 'own authentication'
Verbindung:			
	OLE DB	n	von MapForce nicht offiziell unterstützt
	ODBC	j	
	JDBC	j	via ODBC
	MF used init. Statements	Keine	
	MF used final. Statements	Keine	
SQL:			
	DB-object-name qualification	""	
	support for DB-schemas	j	
	identity support	j	
	MF: read back identity value	identity_val_local()	
	sub select support	j	
	JOIN support	j	
	MF: upper function	UPPER()	
SQL-Ausführung:			
	exec. multiple-stat. in one	n	
	command separator	--	
	special error handling	j	
	retrieve parameter types	n	
	special issues when using ?	n	
Transaktionen:			
	Flat transactions supported	j	
	Start flat-transaction via execution of SQL-command	n	
	Nested transactions supported	n	von DB2 nicht unterstützt
	set transaction isolation	j	
	MF: begin transaction	API-call	
	MF: commit transaction	API-call	
	MF: rollback transaction	API-call	
	MF: set save point	--	von DB2 nicht unterstützt
	MF: rollback to save point	--	von DB2 nicht unterstützt
	MF used init. Statements	Keine	

17.7 Datenbankbeziehungen - beibehalten / aufheben

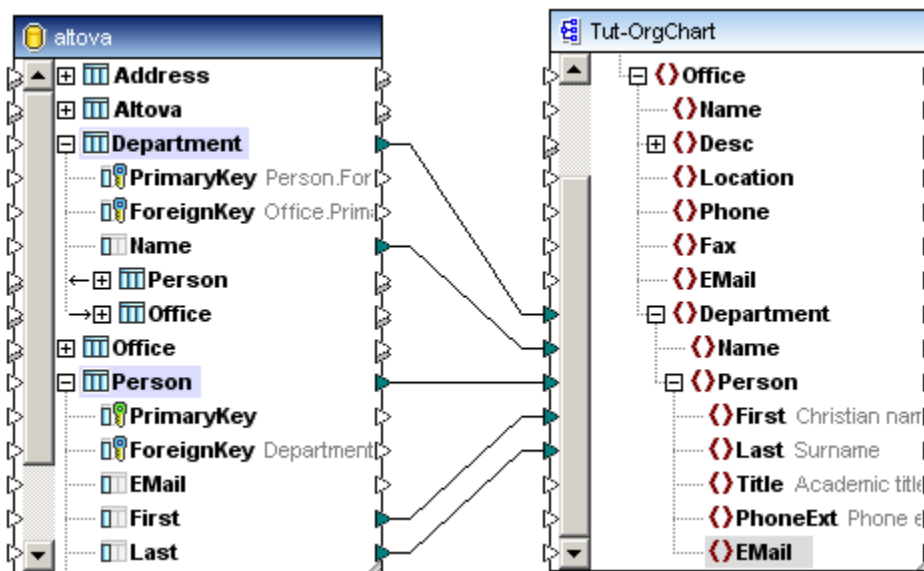
Im folgenden Abschnitt wird beschrieben, wie man Datenbankbeziehungen zwischen Tabellen intakt hält bzw. wie man diese Tabellenbeziehungen aufhebt.

Beibehalten/Verwenden von Datenbankbeziehungen

Um die Beziehung zwischen Datenbanktabellen zu erhalten, müssen Sie Mappings zwischen den verschiedenen Feldern unterhalb einer der **Root**-Tabellen (z.B. Department) erstellen. Unterhalb jeder der "Root"-Tabellen in der Datenbankkomponente wird die vollständige Datenbankstruktur hierarchisch angezeigt.

Nähere Informationen zur hierarchischen Tabellenstruktur von Datenbankkomponenten finden Sie unter [Komponenten und Tabellenbeziehungen](#).

- **Department | Name** wird von der "Root"-Tabelle **Department** aus gemappt.
- **Person | First and Last** werden in der Tabellenhierarchie von der Tabelle **Person** unterhalb der "Root"-Tabelle **Department** gemappt.



Ergebnis des obigen Mappings:

Die Namen der Personen in den einzelnen Abteilungen werden in der Output-Komponenten angezeigt.

```

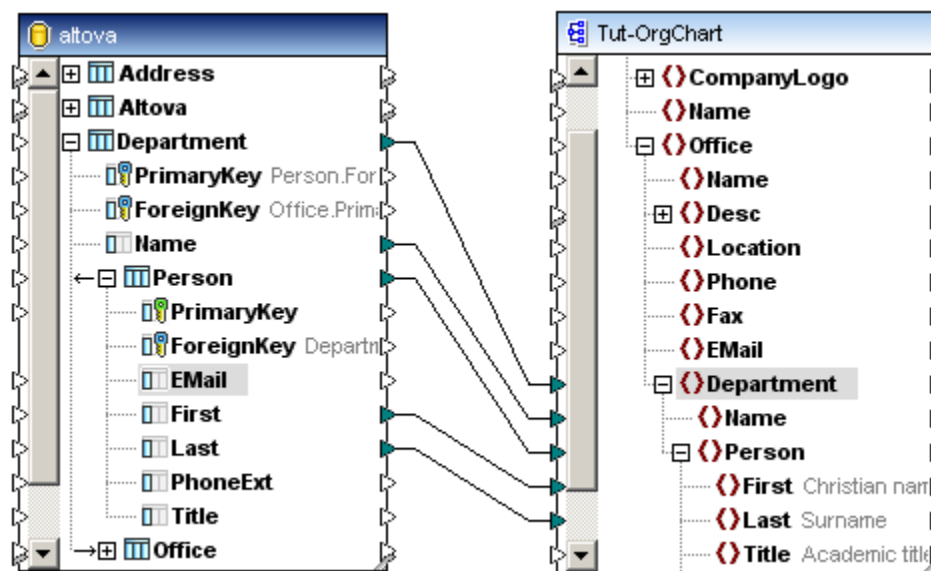
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:noNamespaceSchemaLocation="
   C:/DOCUME~1/MYDOCU~1/Altova/MapForce2010/MapForceExamples
   http://www.w3.org/2001/XMLSchema-instance">
3    <Office>
4      <Department>
5        <Name>Administration</Name>
6        <Person>
7          <First>Vernon</First>
8          <Last>Callaby</Last>
9        </Person>
10       <Person>
11         <First>Frank</First>
12         <Last>Further</Last>
13       </Person>
14       <Person>
15         <First>Loby</First>
16         <Last>Matise</Last>
17       </Person>

```

Aufheben von Datenbankbeziehungen

Erzeugen Sie Mappings derselben Felder aus **verschiedenen** Root-Tabellen (z.B. Department und Office)

- **Department | Name** wird von der "Root"-Tabelle **Department** aus gemappt.
- **Person | First and Last** werden von der Root-Tabelle **Person** gemappt.



Ergebnis des obigen Mappings:

Im Ergebnis werden die Personennamen aller 21 Personen für jedes einzelne Department ausgegeben. Die Tabellenbeziehungen wurden ignoriert. Für jedes einzelne Department werden alle Personen ausgegeben.

Department "Administration".


```
1      <?xml version="1.0" encoding="UTF-8"?>
2      <OrgChart xsi:noNamespaceSchemaLocation="
3          C:/DOCUME~1/MYDOCU~1/Altova/MapForce2010/MapForceExamples
4          http://www.w3.org/2001/XMLSchema-instance">
5          <Office>
6              <Department>
7                  <Name>Administration</Name>
8                  <Person>
9                      <First>Vernon</First>
10                     <Last>Callaby</Last>
11                 </Person>
12                 <Person>
13                     <First>Frank</First>
14                     <Last>Further</Last>
15                 </Person>
16                 <Person>
17                     <First>Loby</First>
18                     <Last>Matise</Last>
19                 </Person>
20             </Department>
21         </Office>
22     </OrgChart>
```

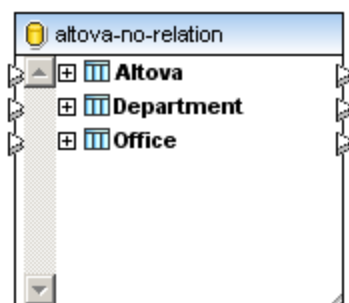
17.8 Lokale Beziehungen - Erzeugen von Datenbankbeziehungen

MapForce erlaubt das Extrahieren verwandter Datenbankdaten, selbst wenn in der Quelldatenbank keine derartigen Beziehungen existieren. Sie können die Primär-/Sekundärschlüsselbeziehungen zwischen Tabellen in einer Komponente definieren, ohne dadurch die zugrunde liegenden Datenbankbeziehungen in irgendeiner Weise zu beeinflussen.

Diese on-the-fly-Beziehungen werden in MapForce **lokale Beziehungen** genannt.

- Jedes Datenbankfeld kann als Primär- oder Sekundärschlüssel verwendet werden
- Es können neue Beziehungen erzeugt werden, die derzeit in der Datenbank nicht existieren

Die in diesem Beispiel verwendete MS Access-Datenbank **altova-no-relation.mdb** ist eine vereinfachte Version der mit MapForce zur Verfügung gestellten Datenbank **Altova.mdb**. In MS Access wurden die Tabellen **Person** und **Address** sowie die Tabellenbeziehungen entfernt.



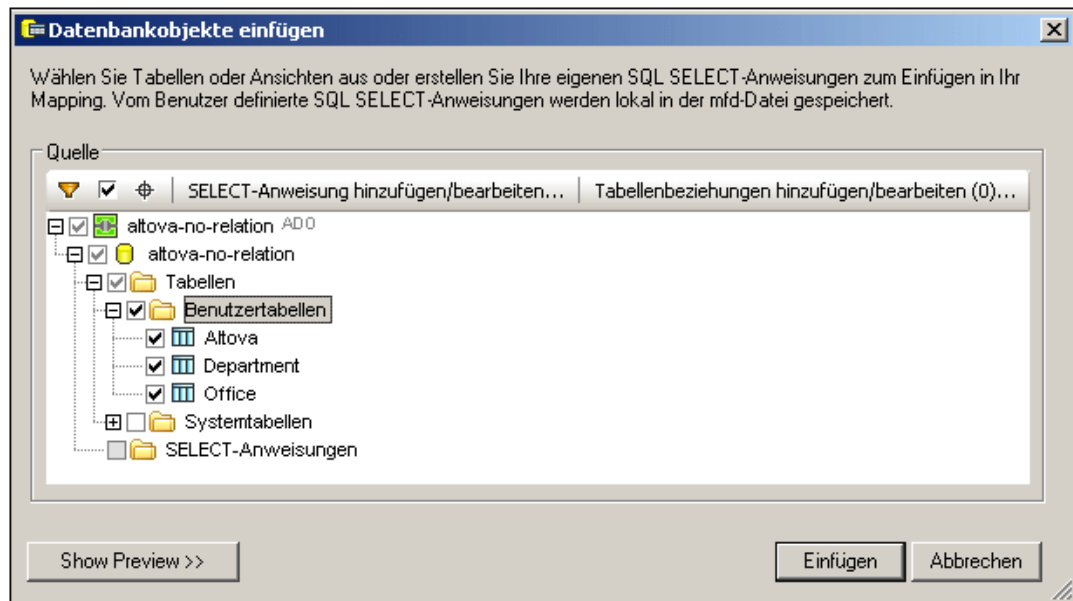
Ziel dieses Beispiels ist es, die Niederlassungen von Altova und die einzelnen Abteilungen darin anzuzeigen.

Keine der Tabellen in der **altova-no-relation**-Struktur hat Child-Elemente, alle Tabellen befinden sich auf derselben "**Root**"-Ebene. Der Inhalt der einzelnen Tabellen beschränkt sich auf die darin enthaltenen Felder. Mit Hilfe von MapForce können wir dennoch verwandte Datenbankdaten extrahieren, obwohl Beziehungen nicht explizit definiert sind.

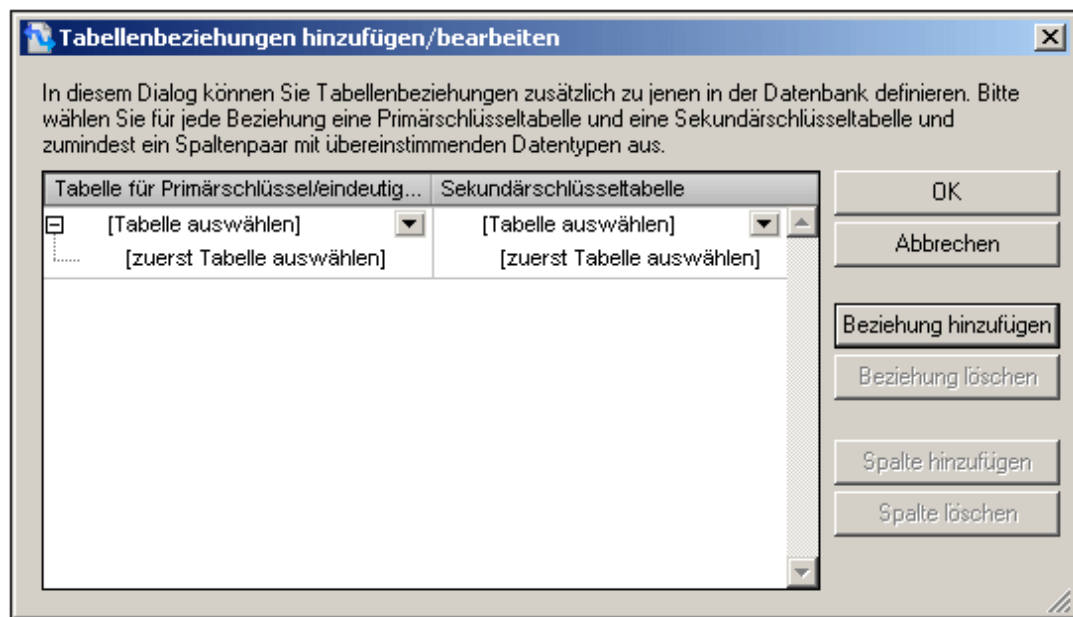
So erstellen Sie lokale Beziehungen:

Lokale Beziehungen können durch Einfügen einer Datenbank oder durch Rechtsklick auf eine bestehende Datenbankkomponente und Auswahl des Befehls "Tabellen hinzufügen/entfernen" aus dem Kontextmenü definiert werden.

1. Fügen Sie die Datenbank **altova-no-relation** ein.
2. Wählen Sie im Verbindungsassistenten Microsoft Access aus und klicken Sie auf "Weiter".
3. Klicken Sie auf "Durchsuchen", wählen Sie **Altova.mdb** und aktivieren Sie das Kontrollkästchen **Benutzertabellen**, um alle drei Tabellen auszuwählen.



4. Klicken Sie in der Symbolleiste auf die Schaltfläche **Tabellenbeziehungen hinzufügen/bearbeiten**.
Daraufhin wird das Dialogfeld "Tabellenbeziehungen hinzufügen/bearbeiten" geöffnet.
5. Klicken Sie auf die Schaltfläche **Beziehung hinzufügen**.

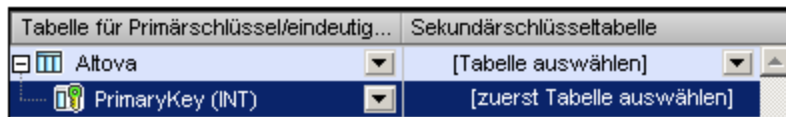


Über die beiden Auswahllisten können Sie die Tabellen auswählen, für die Beziehungen erstellt werden sollen. Die linke Auswahlliste ist die Tabelle für Primärschlüssel/eindeutige Schlüsse, die rechte die Sekundärschlüsseltabelle.

6. Klicken Sie auf die linke Auswahlliste und wählen Sie die Tabelle **Altova** aus.



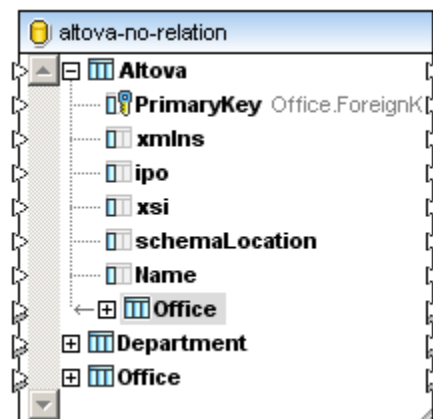
7. Klicken Sie auf die Auswahlliste "select column" unterhalb und wählen Sie den Eintrag **Primary Key**.



8. Wählen Sie als Sekundärschlüsseltabelle die Tabelle **Office** und die Spalte **ForeignKey** aus.



9. Klicken Sie auf die Schaltfläche OK, um die Definition der lokalen Beziehung abzuschließen und klicken Sie anschließend auf OK, um die Datenbank in den Mapping-Bereich einzufügen.

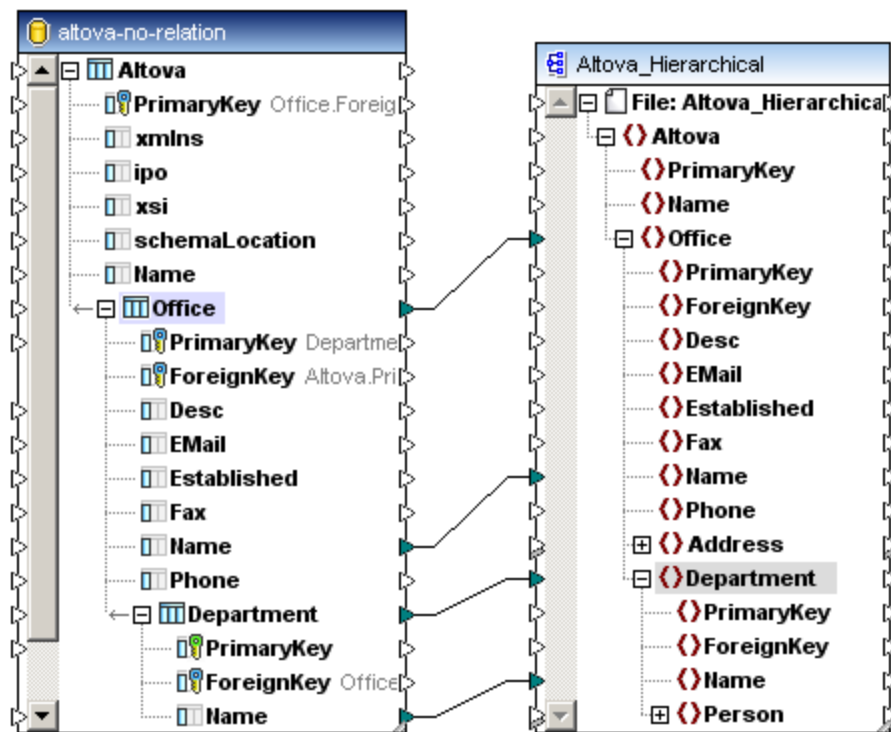


Wenn Sie auf das Erweiterungssymbol der Tabelle "Altova" klicken, sehen Sie, dass es eine Beziehung zwischen Altova und den Office-Tabellen gibt. Die Office-Tabelle wird unterhalb der Tabelle "Altova" als verwandte Tabelle mit ihrem eigenen Erweiterungssymbol angezeigt.

10. Mit derselben Methode können Sie eine Beziehung zwischen den Tabellen **Office** und **Department** herstellen.
Klicken Sie mit der rechten Maustaste auf die Datenbankkomponente und wählen Sie im Kontextmenü den Befehl **Tabellen hinzufügen/entfernen**, klicken Sie anschließend in der Symbolleiste auf die Schaltfläche "Tabellenbeziehung hinzufügen/bearbeiten".



Um beim Mapping Beziehungen zwischen Tabellen beizubehalten, verwenden Sie Verbindungen unterhalb **einer** der Root-Tabellen, d.h. in diesem Fall Altova.



Nachdem Sie wie oben gezeigt Mappings definiert haben, wechseln Sie zum Register "Ausgabe" um sofort ein Ergebnis des Mappings zu sehen. Datenbankdaten können nicht in der Vorschau angezeigt werden, wenn die Zielsprache XSLT, XSLT2 oder XQuery ist. Es erscheint dann eine entsprechende Warnmeldung und die Datenbankkomponente wird ausgegraut angezeigt.

Im Mapping-Ergebnis wird nun folgendes angezeigt:

- "für jedes Office-Element sollen der Büroname und anschließend alle Abteilungen in diesem Büro ausgegeben werden"

```
<Altova xsi:noNamespaceSchemaLocation="C:/DOCUME~1/MYDOCU~1/Altov
<Office>
  <Name>Nanonull, Inc.</Name>
  <Department>
    <Name>Administration</Name>
  </Department>
  <Department>
    <Name>Marketing</Name>
  </Department>
  <Department>
    <Name>Engineering</Name>
  </Department>
  <Department>
    <Name>IT & Technical Support</Name>
  </Department>
</Office>
<Office>
  <Name>Nanonull Partners, Inc.</Name>
  <Department>
    <Name>Administration</Name>
```

17.9 Mappen großer Datenbanken mit Hilfe von MapForce

Bei Verwendung von Datenbanken mit vielen Tabellen in einem Mapping zeigt MapForce für die gesamte Datenbank alle Datenbankbeziehungen zwischen den importierten Tabellen an. Dies liegt daran, dass die Applikation nicht automatisch weiß, welche Tabellen für das Mapping verwendet werden. Daher müssen alle Möglichkeiten in Betracht gezogen werden.

Wenn alle Tabellen in einer einzigen Datenbankkomponente ausgewählt sind, kann dies zu allzu großen Datenstrukturen führen. Sie können jedoch mehrere Datenbankkomponenten derselben Datenbank erstellen, in der/die nur die Tabellen benutzt/importiert werden, die für das Mapping benötigt werden. Auf diese Art wird das Mapping auch intuitiver.

Beispiel:

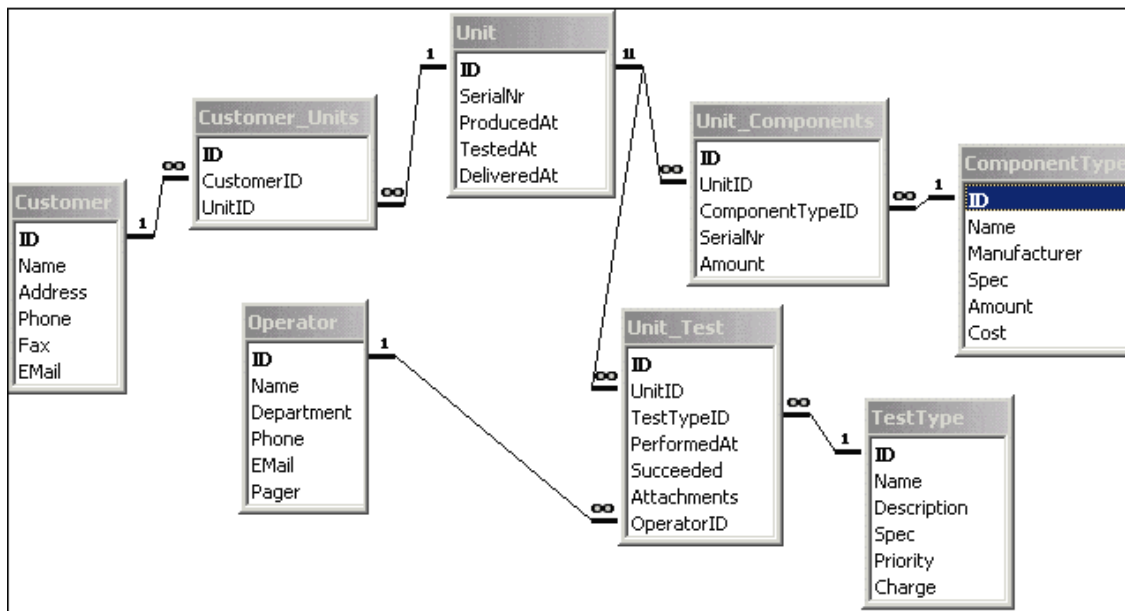
In einer Produktionsfirma werden verschiedene Komponenten zu kundenspezifischen Geräten zusammengebaut. Bevor das Gerät geliefert wird, werden die Geräte getestet und die Ergebnisse werden in einer Datenbank gespeichert.

Zu einem bestimmten Zeitpunkt während der Testphase des Prototyps stellt sich heraus, dass eine Charge von Komponenten defekt ist, woraufhin ein Rückruf für diese Produkte gestartet wird. Ziel des Mappings ist die Generierung einer Liste aller betroffenen Kunden, die schriftlich verständigt werden müssen.

In diesem Fall definiert das Mapping:

für den ComponentType name = "Prototype" UND den Manufacturer (Hersteller) = "Noname",
Wähle alle damit in Beziehung stehenden Kunden mit den benötigten Informationen.

Unten sehen Sie das Diagramm der Beziehungen der in diesem Abschnitt verwendeten Beispieldatenbank.



Im Ordner MapForceExamples finden Sie ein SQL Script (CreateProductionsTables.sql) zum Erstellen der Datenbank in SQL Server 2008.

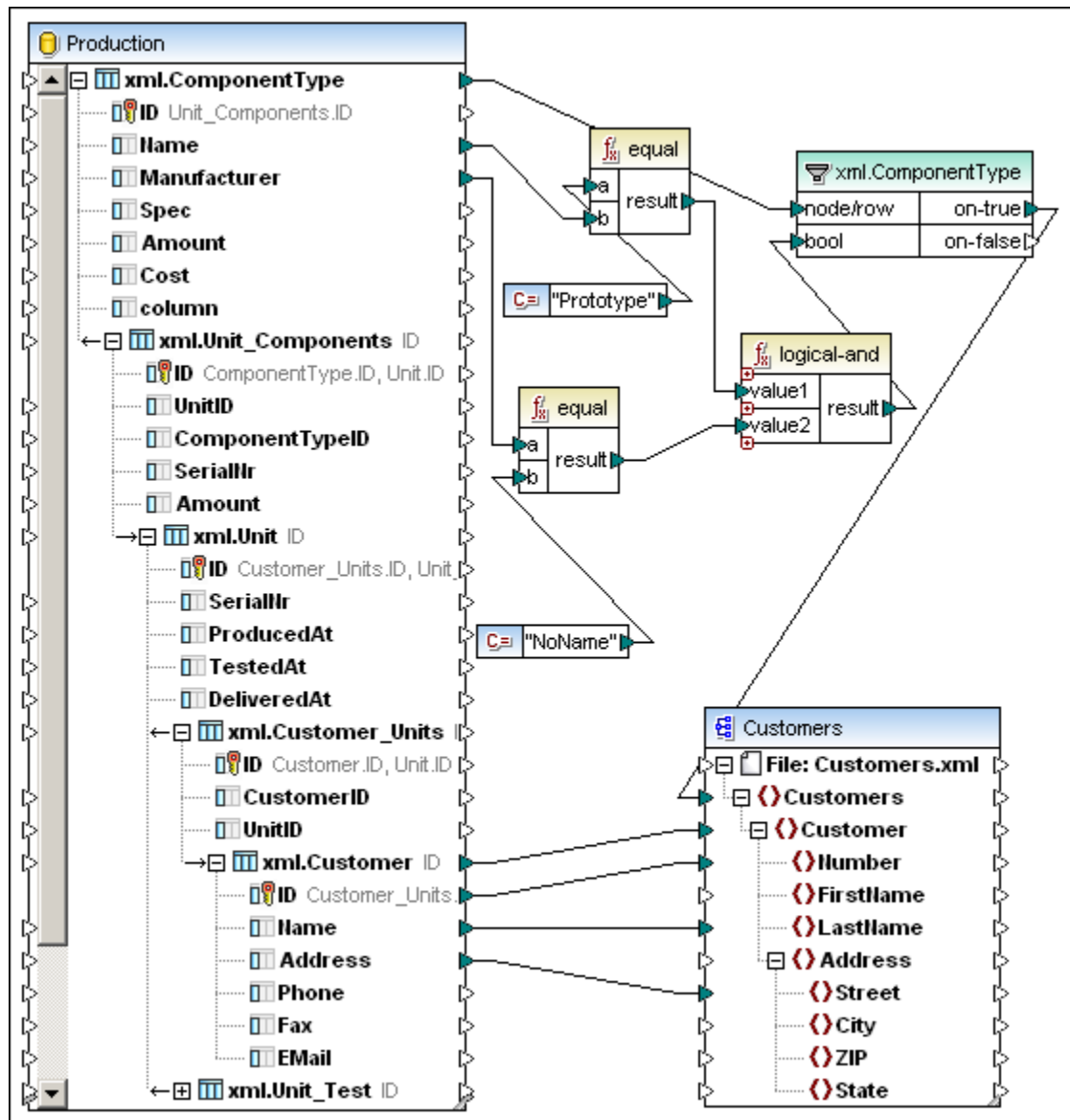
17.9.1 Import der gesamten Datenbank

Option 1:

Importieren Sie die gesamte Datenbank, also die Datenbank mit **allen** ihren Tabellen. Die Datenbankkomponente "Production" enthält daher alle Tabellen, wobei jede Tabelle als "Root"-Tabelle mit ihren dazugehörigen Tabellen angezeigt wird.

Bei Verwendung der ComponentType-Tabelle als Root-Tabelle wird beim Mapping nach folgenden Kriterien gefiltert:

- dem Komponentennamen "Prototype" UND
- dem Hersteller (Manufacturer) "NoName" sowie der
- dazugehörigen Kunden-ID und den Adresdaten



17.9.2 Datenbank-Teilimport

Option 2:

Importieren Sie nur die Tabellen, die für das Extrahieren der benötigten Informationen erforderlich sind, also:

- Selektieren aller defekten Geräte
- Selektieren aller Kunden, an die diese Geräte geliefert wurden

Fügen Sie zwei Datenbankkomponenten aus derselben Datenbank ein, die unterschiedliche Tabellen importieren.

Komponente 1: Fügen Sie die folgenden Tabellen ein:

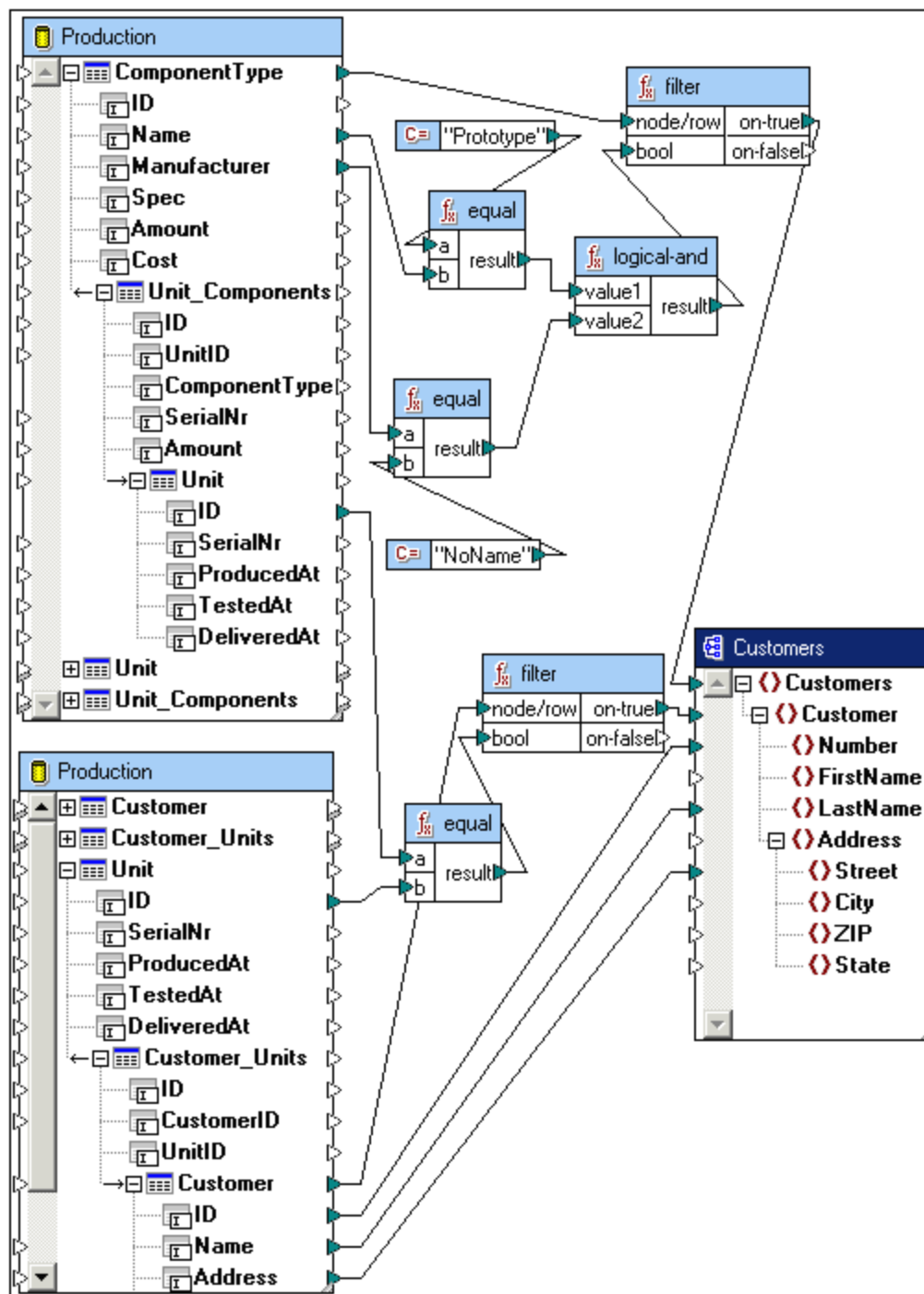
- ComponentType
- Unit_Components
- Unit

Komponente 2: Fügen Sie ein:

- Unit
- Customer_Units
- Customer

Mapping:

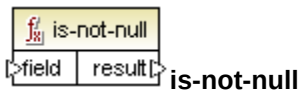
- Filtern Sie den Komponentennamen "Prototype" UND den Herstellernamen (Manufacturer) "NoName" (Komponente 1).
- Verwenden Sie die "equal"-Funktion, um die Geräte-ID aus Komponente 1 mit der Geräte-ID aus Komponente 2 zu vergleichen.
- Wenn die IDs identisch sind, übergeben Sie die dazugehörigen Kundendaten mittels der Filterkomponente von Komponente 2 an die XML-Datei "Customers".



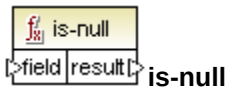
17.10 Datenbank, Null-Verarbeitungsfunktionen

Die DB-Language-Bibliothek wurde um neue Null-Verarbeitungsfunktionen erweitert.

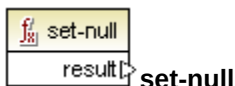
 db	
is-not-null	result = is-not-null(field)
is-null	result = is-null(field)
set-null	result = set-null()
substitute-null	result = substitute-null(field, replace-with



Gibt "false" zurück, wenn das Feld Null ist, andernfalls wird "true" zurückgegeben.



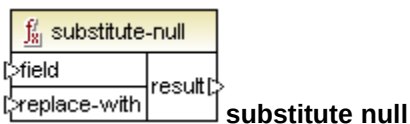
Gibt "true" zurück, wenn das Feld Null ist, andernfalls wird "false" zurückgegeben.



Damit wird eine Datenbankspalte auf Null gesetzt. Diese Funktion überschreibt auch einen Standardwert mit Null. Bei Verbindung mit einem anderen Node, also einem Node, bei dem es sich nicht um ein Datenbankfeld handelt, so fungiert diese Funktion als leere Sequenz.

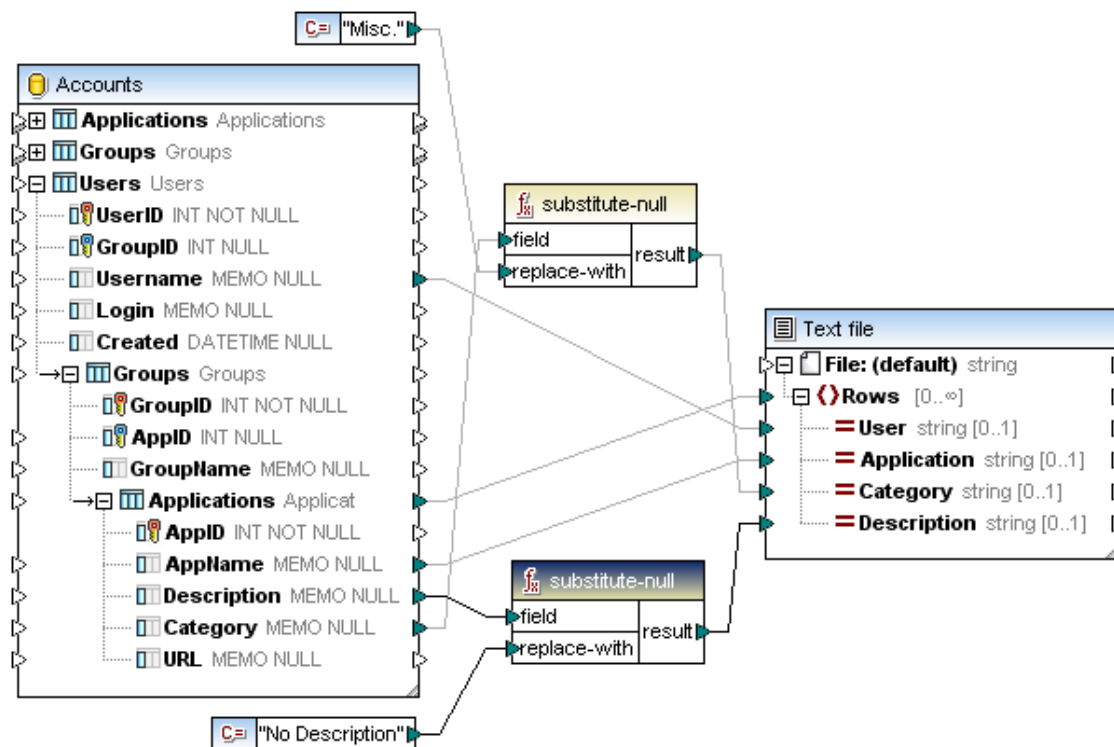
Bitte beachten Sie:

- Wenn Sie diese set-null mit einer anderen Funktion verbinden, hat dies normalerweise zur Folge, dass die andere Funktion gar nicht aufgerufen wird. Wenn Sie set-null mit einer Sequenzfunktion verbinden, z.B. count, so wird die Funktion mit einer leeren Sequenz aufgerufen.
- Bei Verbindung mit Sonderfunktionen, Filtern und IF-Else-Bedingungen werden die Felder wie erwartet auf Null gesetzt.
- Bei Verbindung von set-null mit einem simpleType-Element wird dieses Element in der Zielkomponente nicht erstellt.
- Das Verbinden von set-null mit einem complexType-Element, einer Tabelle oder Zeile ist nicht zulässig. In diesem Fall wird ein Validierungsfehler angezeigt.



Damit wird der aktuelle Feldinhalt, falls vorhanden, gemappt. Andernfalls wird das Datenelement verwendet, das auf den replace-with-Parameter gemappt ist.

In der Abbildung unten sehen Sie ein Beispiel für die Verwendung der substitute-null Funktion. Sie finden das Beispiel im Ordner ...**MapForceExamples** unter dem Namen "**DB-ApplicationList**".



Die erste Funktion überprüft, ob in der Tabelle "Applications" ein Eintrag "Category" vorhanden ist. Da für die Applikation "Notepad" kein solcher Eintrag vorhanden ist, wird in der Textdatei "Misc" auf das Datenelement "Category" gemappt.

Die zweite Funktion überprüft, ob ein Eintrag "Description" vorhanden ist und mappt den String "No description", falls dies wie bei der Applikation "Notepad" nicht der Fall ist.

17.11 SQL WHERE/ORDER-Komponente

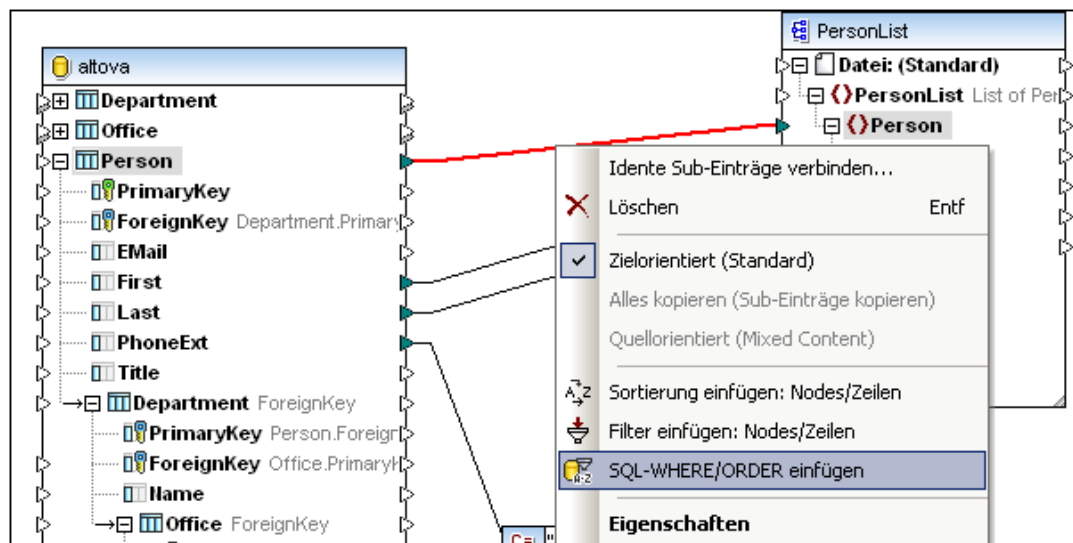
Sie können in MapForce Datenbankdaten mit Hilfe der SQL WHERE/ORDER-Komponente filtern und sortieren. Das hier erläuterte Beispiel steht unter dem Namen **DB_PhoneList.mfd** im Ordner `..\MapForceExamples` zur Verfügung.

Die SQL WHERE/ORDER-Komponente besteht aus mehreren Teilen:

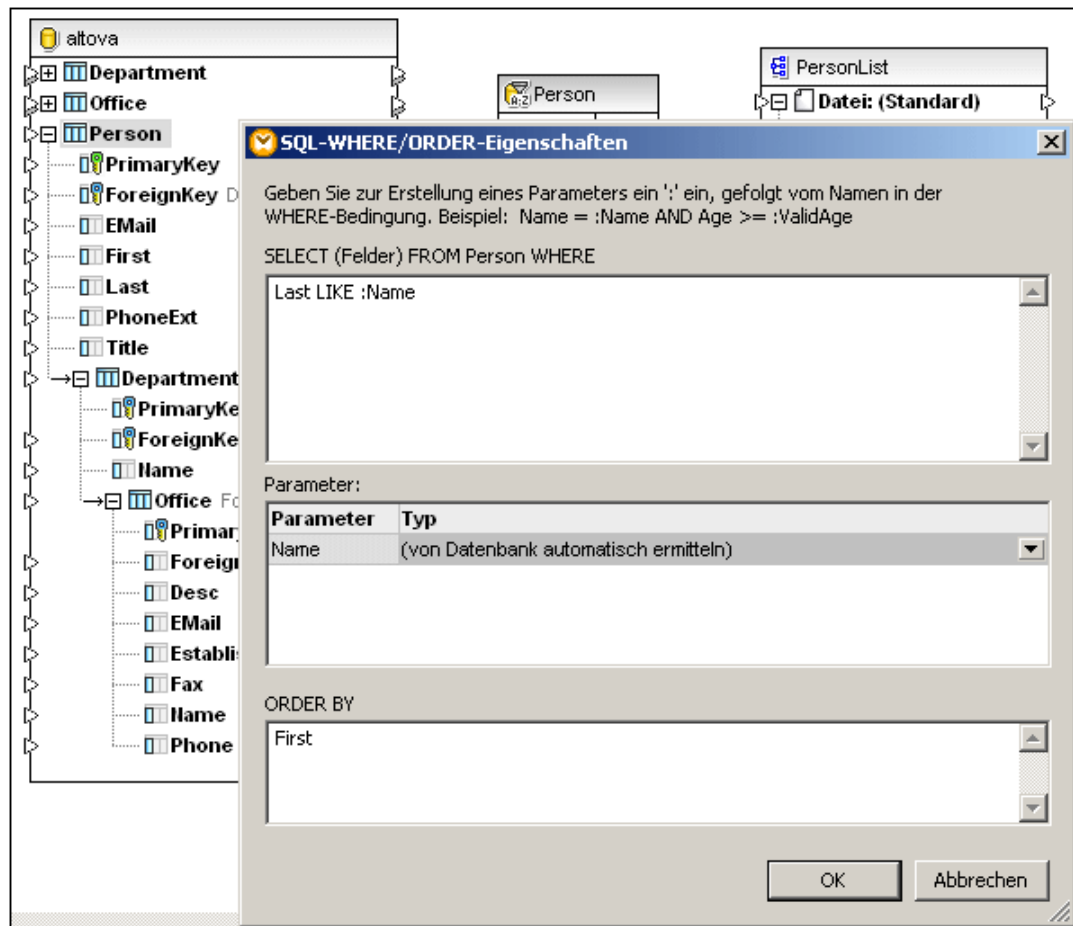
- der **SELECT** -Anweisung, die automatisch generiert wird, wenn eine Verbindung zu einer Datenbanktabelle oder einem Datenbankfeld hergestellt wird
- der **WHERE**-Klausel, die Sie manuell in das SELECT-Textfeld eingeben. Beachten Sie, dass die Sekundärschlüssel automatisch in die Select-Anweisung inkludiert werden.
- der **ORDER BY**-Klausel

So fügen Sie eine SQL WHERE/ORDER-Komponente ein:

1. Klicken Sie mit der rechten Maustaste auf einen Konnektor zwischen einer Tabelle und dem Ziel-Node.



2. Wählen Sie im Kontextmenü den Befehl "SQL-WHERE/ORDER einfügen". Daraufhin wird das Dialogfeld "SQL-WHERE/ORDER-Eigenschaften" geöffnet.
3. Klicken Sie in das oberste Feld, um die WHERE-Abfrage in das Textfeld einzugeben. Die SELECT-Anweisung genau oberhalb des Textfelds wird automatisch für Sie generiert, wenn Sie eine Verbindung zwischen einer Tabelle, einem Feld oder dem **Tabellen/Feld-Input-Symbol** der SQL WHERE/ORDER-Komponente erstellen.

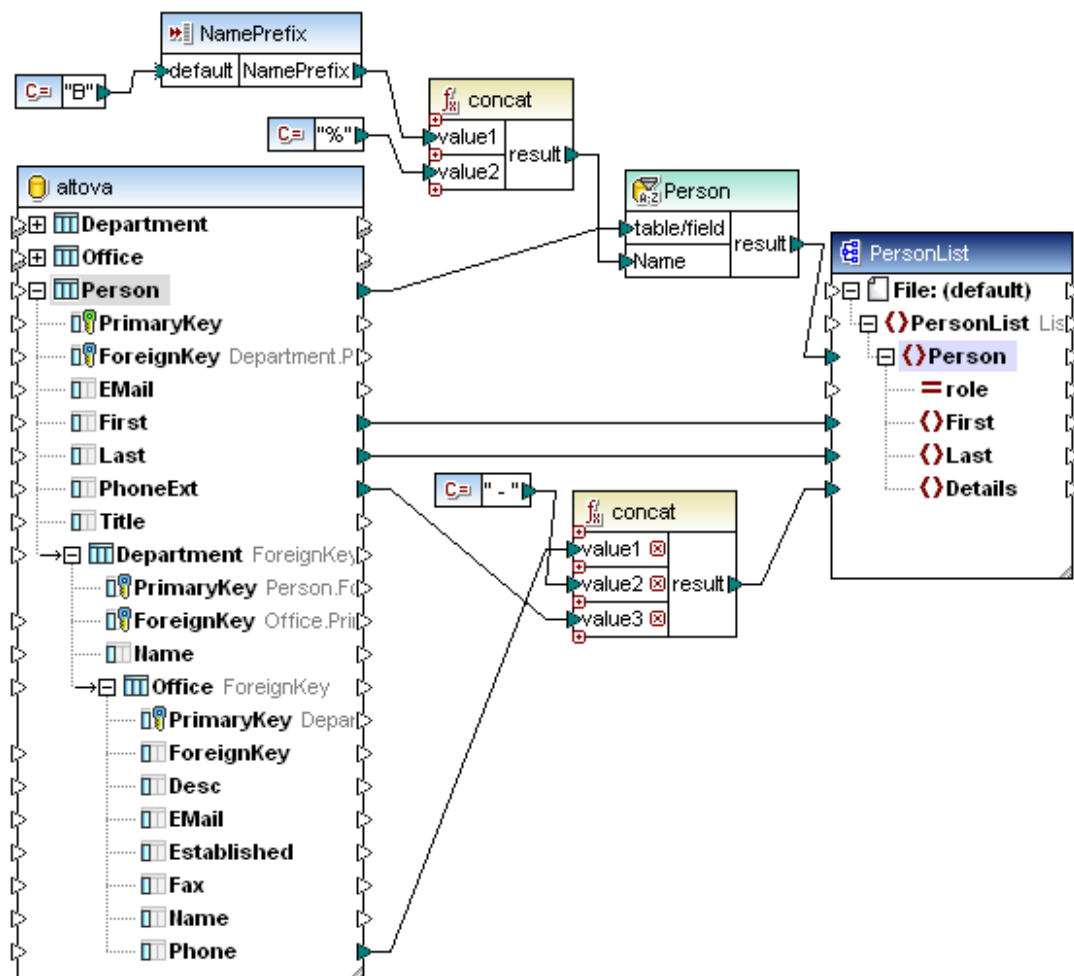


Mit der WHERE-Anweisung im SELECT-Textfeld wird ein Parameter "Name" definiert, der mit Hilfe des Schlüsselworts LIKE ein Pattern im Feld "Last" der Tabelle "Person" sucht. In diesem Fall liefert eine Konstantenkomponente den Suchstring **B**, mit dem alle Personen, deren Nachname mit "B" beginnt, gefunden werden.

Das Platzhalterzeichen "%" (in der Konstantenkomponente) steht für beliebig viele Zeichen.

Im Bereich "Parameter" können Sie den Datentyp eines in der Abfrage definierte Parameters definieren (solange die Komponente mit keiner Datenbank verbunden ist, wird im Dialogfeld eine Warnmeldung angezeigt).

4. Geben Sie in den Bereich ORDER BY "First" ein, damit die Ergebnisdatensätze nach dem Vornamen der Person ("First") geordnet werden.

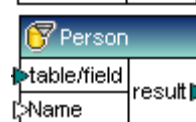


Bitte beachten Sie Folgendes:

Das in der SQL-WHERE/ORDER-Titelleiste angezeigte Symbol zeigt an, welche Parameter definiert wurden:



Es wurde eine WHERE-Klausel definiert, z.B. First > "C" AND Last > "C"



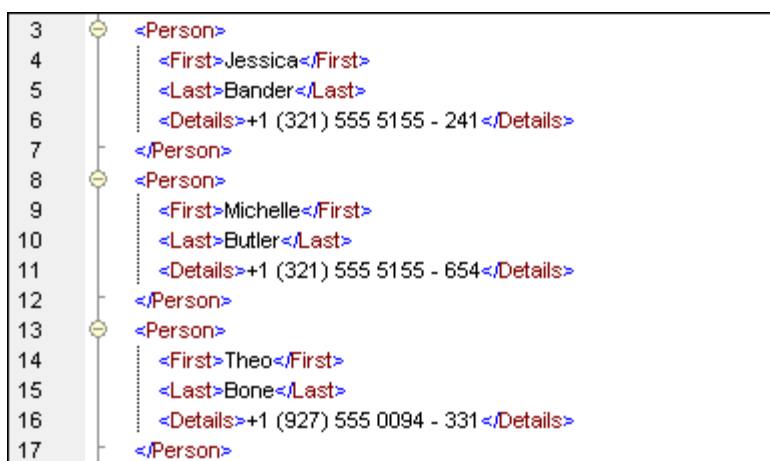
Es wurde eine WHERE-Klausel definiert, z.B. Last LIKE :Name
Der definierte Parameter "Name" ist unterhalb des Parameters "table/field" zu sehen.



Es wurde eine WHERE-Klausel definiert, z.B. Last LIKE :Name
Zusätzlich dazu wurde eine ORDER BY-Klausel definiert, z.B. ORDER BY Last. Die Sortierreihenfolge wird durch das Sortiersymbol A-Z angezeigt.

Wenn Sie den Mauszeiger über die Titelleiste von SQL WHERE/ORDER platzieren, wird ein Popup-Fenster mit den verschiedenen definierten Klauseln angezeigt.

Alle Personen, deren Nachnamen mit "B" beginnen, werden nun auf dem Ausgaberegister nach ihrem Vornamen sortiert.



17.11.1 SQL WHERE/ORDER-Operatoren

In der WHERE-Komponente werden die folgenden Operatoren unterstützt:

Operator	Beschreibung
=	Ist gleich
<>	Ist nicht gleich
<	Kleiner als
>	Größer als
>=	Größer oder gleich
<=	Kleiner oder gleich
IN	Ruft einen bekannten Wert einer Spalte ab
LIKE	Suche nach einem bestimmten Muster
BETWEEN	Suche innerhalb eines Bereichs

Platzhalterzeichen:

Das Platzhalterzeichen % steht für beliebig viele Zeichen in einem Muster (entspricht dem Zeichen * in anderen Programmen) z.B. ruft %r alle Datensätze ab, die mit "r" enden.

XML-Daten:

XQuery-Befehle werden auch bei der Abfrage von Datenbanken unterstützt, die das Speichern und die Abfrage von [XML-Datenbankdaten](#) unterstützen, z.B. IBM DB2.

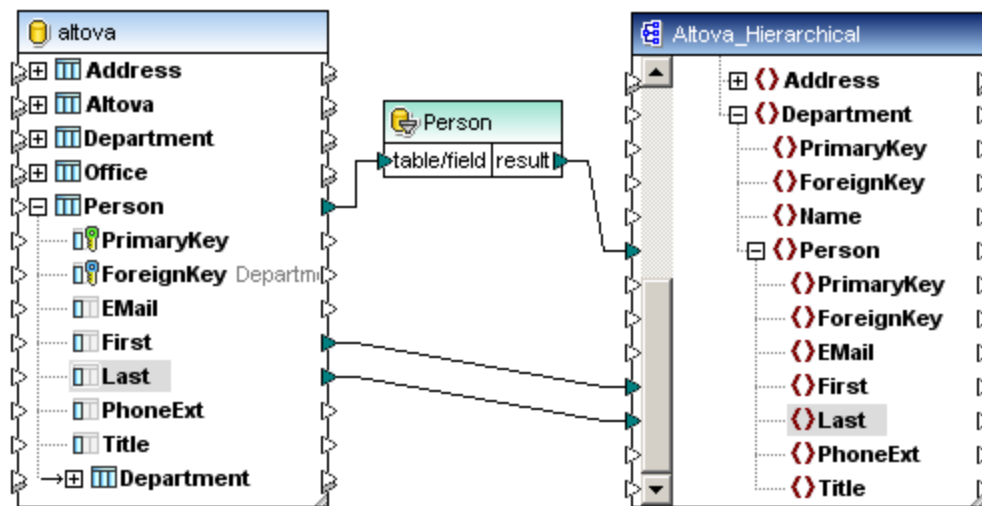
Beispiel **xmlexists**('\$c/Client/Address[zip>"55116"]' passing USER.CLIENTS.
CONTACTINFO AS "c")

Select * FROM Person WHERE
First > "C" AND Last > "C"

Ruft diejenigen Datensätze ab, deren Inhalt von First und Last größer ist als der Buchstabe C.
Ruft alle Namen ab Callaby (aus altova.mdb) ab.

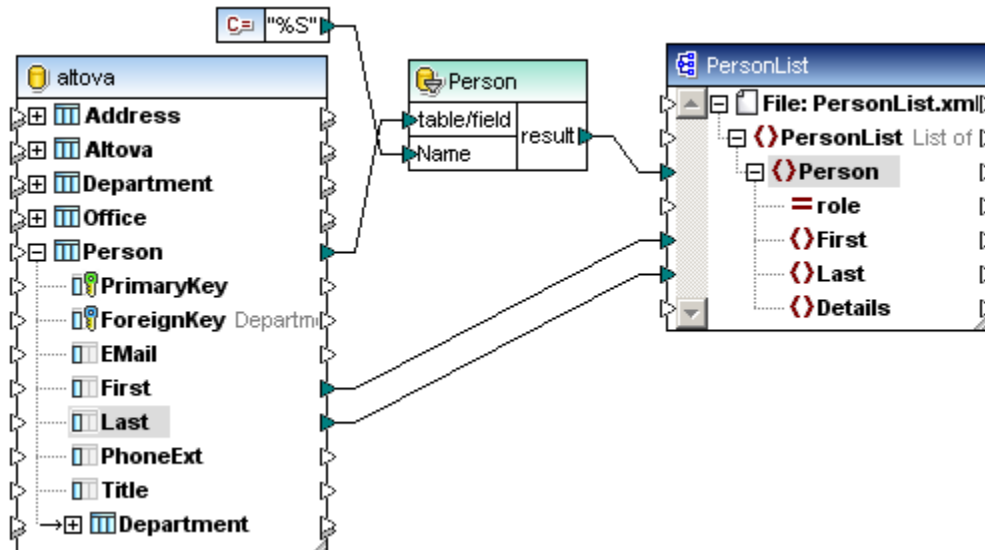
Beachten Sie, wie die Konnektoren platziert wurden:

- Der Konnektor zum Parameter **table/field** ist mit der Tabelle verbunden, an der Sie die Abfrage durchführen möchten, in diesem Fall als "Person".
- Der Parameter **result** ist mit einem übergeordneten Datenelement der Felder verbunden, die abgefragt/gefiltert werden, in diesem Fall mit dem Datenelement "Person". Die Felder "first" und "last" sind mit Subelementen in der Zielkomponente verbunden, die im Ergebnis aufscheinen sollen.



Select from Person WHERE Last LIKE :Name

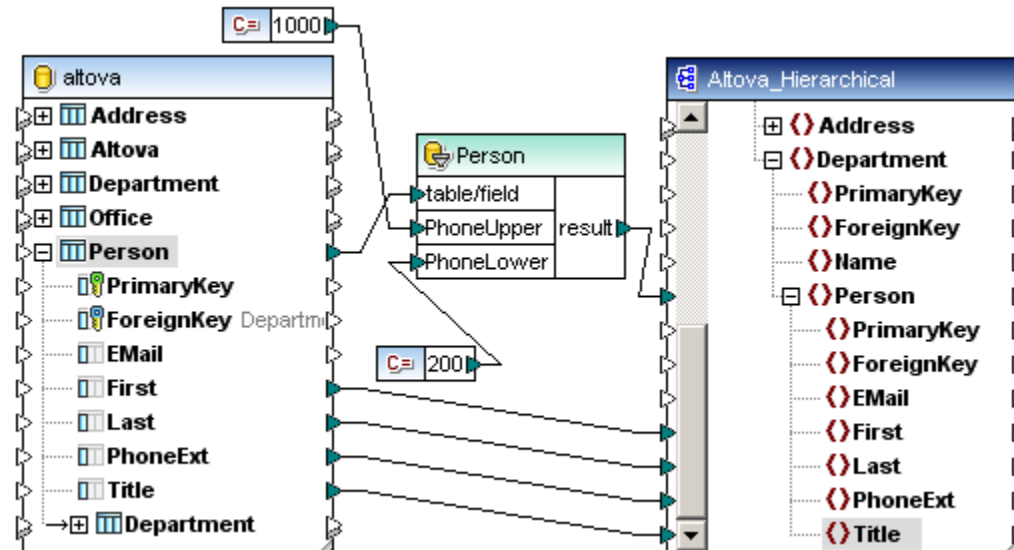
Mit dieser WHERE-Klausel wird ein Parameter **Name** erstellt, der als Parameter in der SQL WHERE/ORDER-Komponente aufscheint. In diesem Fall dient er zum Suchen eines Musters in der Spalte "Last". Das Platzhalterzeichen % steht für eine beliebige Anzahl von Zeichen.



Die Konstantenkomponente liefert die Werte für die Suche nach allen Nachnamen, die mit dem Buchstaben "S" enden, d.h. %S.

Select from Person WHERE PhoneExt < :PhoneUpper and PhoneExt > :PhoneLower

Mit dieser WHERE-Anweisung werden zwei Parameter erstellt, PhoneUpper und PhoneLower, mit denen die aktuellen Werte von PhoneExt verglichen werden. Der obere und der untere Wert stammen aus den zwei Konstantenkomponenten aus dem unten gezeigten Diagramm.



Bitte beachten Sie:
Die WHERE-Klausel in diesem Beispiel könnte auch mit Hilfe des Operators BETWEEN neu erstellt werden.

Select from Person WHERE
PhoneExt BETWEEN :PhoneUpper and :PhoneLower

17.12 Mappen von XML-Daten von / auf Datenbanken

MapForce unterstützt derzeit das Mappen von XML-Daten von / auf verschiedene Datenbanken, wie z.B. MS Access und IBM DB2 Version 9. In diesem Abschnitt wird das Mapping der Datei **xml2access.mfd** aus dem Ordner ...\\Tutorial beschrieben.

XML-Dokumente können von/auf alle **string**, **varchar** und **memo**-Felder mit ausreichender Länge gemappt werden. Bitte beachten Sie, dass die Zeichenkodierung derartiger Dokumente immer die des zugrunde liegenden String-Felds in der jeweiligen Datenbank ist. Wenn der Text im Feld nicht als Unicode gespeichert ist, können manche Zeichen nicht dargestellt werden. Eine vollständige Unterstützung für alle XML-Kodierungen ist nur in nativen Feldern von IBM DB2 Version 9 Datenbanken möglich.

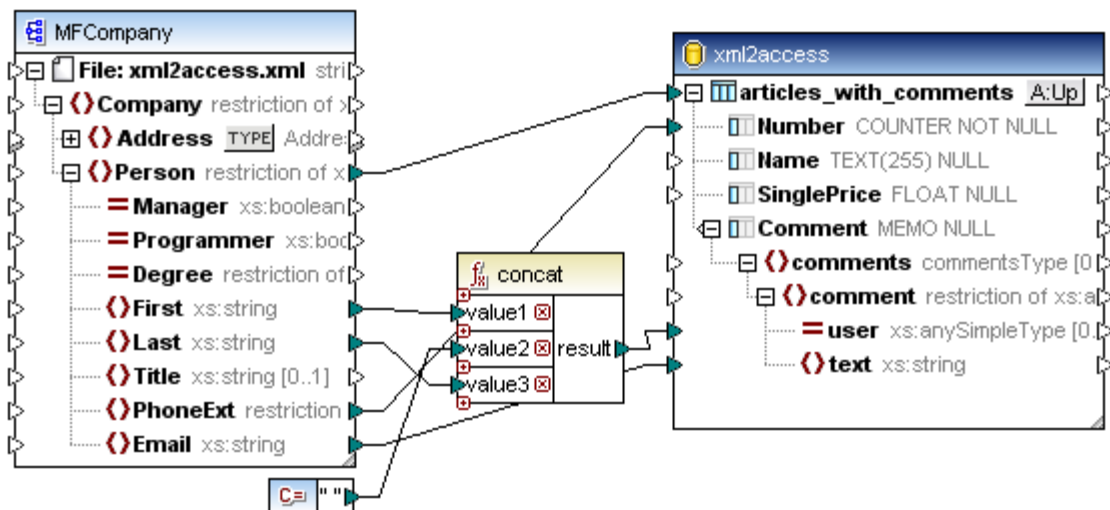
Damit die XML-Nodes in der Datenbankkomponente, die das Mappen von und auf die XML-Elemente gestattet, angezeigt werden, muss ein Schema **zugewiesen werden**. Nachdem dies durchgeführt wurde, wird das Mapping wie gewöhnlich fortgesetzt.

MapForce unterstützt derzeit die Zuweisung **eines** XML-Schemas pro Datenbankspalte und die Auswahl eines **einigen** Root-Elements dieses Schemas.

Öffnen Sie die Mapping-Datei **xml2access.mfd** im Ordner ...\\Tutorial.

Zweck dieses Mappings:

- Die XML-Daten in der **Zieldatenbank** in der Spalte **Comment** werden aktualisiert, wenn:
- der Inhalt von Number der Felder PhoneExt und Number identisch ist.

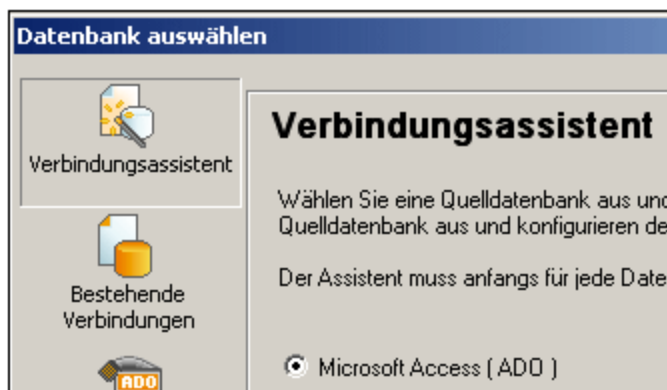


So fügen Sie die Datenquellenkomponente ein:

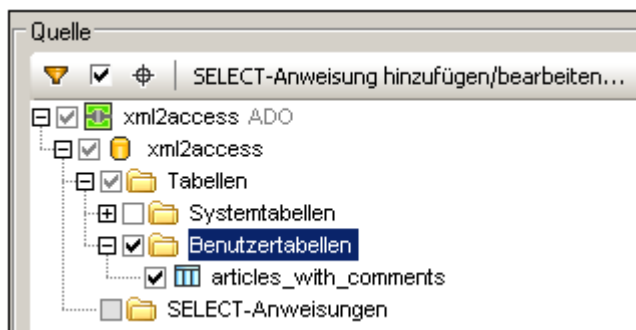
1. Klicken Sie auf die Schaltfläche **XML-Schema/Datei einfügen** , und wählen Sie das Schema **MFCCompany.xsd** aus.
2. Klicken Sie auf "Ja", wenn Sie aufgefordert werden, eine **XML-Beispieldatei** anzugeben und wählen Sie **xml2access.xml** aus.

So fügen sie das Datenbankziel ein und mappen Daten darauf:

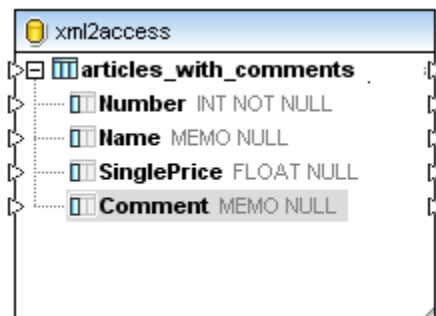
1. Klicken Sie in der Symbolleiste auf die Schaltfläche **Datenbank einfügen** , klicken Sie auf MS Access (ADO) und anschließend auf "Weiter".



2. Klicken Sie auf die Schaltfläche "Durchsuchen" und wählen Sie die Datei **xml2access.mdb** im Ordner ...**Tutorial** aus. Klicken Sie anschließend auf "Weiter".

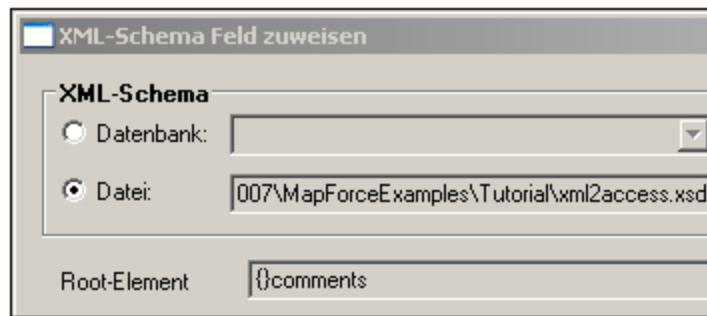


3. Aktivieren Sie das Kontrollkästchen der Tabelle **articles_with_comments** und klicken Sie auf OK.

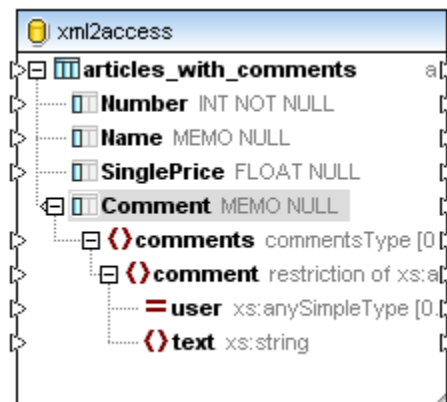


So weisen Sie einer Datenbank ein XML-Schema zu:

1. Rechtsklicken Sie in der Datenbank-Zielkomponente auf den Eintrag/die Spalte "Comment" und wählen Sie den Befehl **XML-Schema Feld zuweisen...**
2. Aktivieren Sie, falls notwendig, das Optionsfeld **Datei**, wählen Sie **xml2access.xsd** und klicken Sie auf OK.

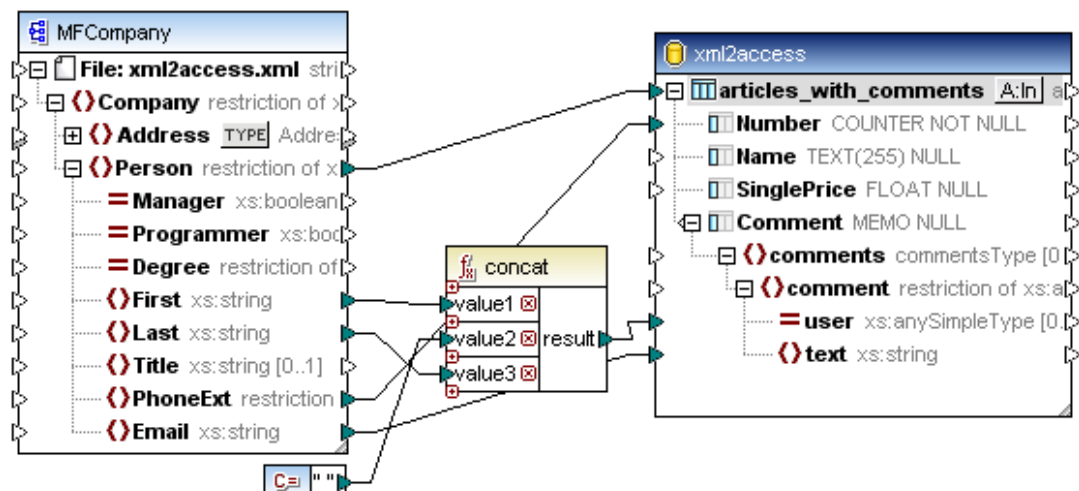


- Erweitern Sie den Eintrag **Comment**, um Konnektoren zu den entsprechenden Einträgen zu erstellen.



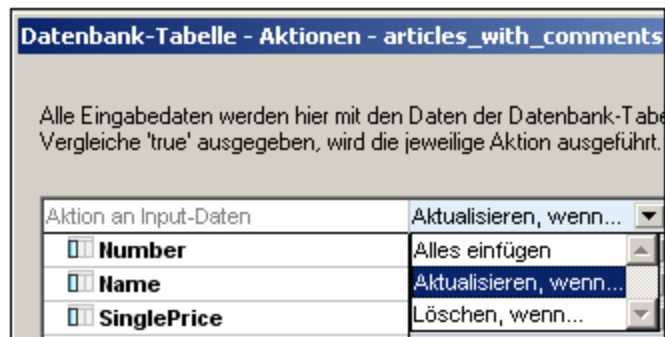
Die Struktur der XML-Schema-Nodes mit allen Einträgen, die gemappt werden können, wird unterhalb des Eintrags/der Spalte "Comment" angezeigt. Sie können nun XML-Daten auf die Datenbank mappen.

- Erstellen Sie die Mapping-Konnektoren wie in der Abbildung oben gezeigt.

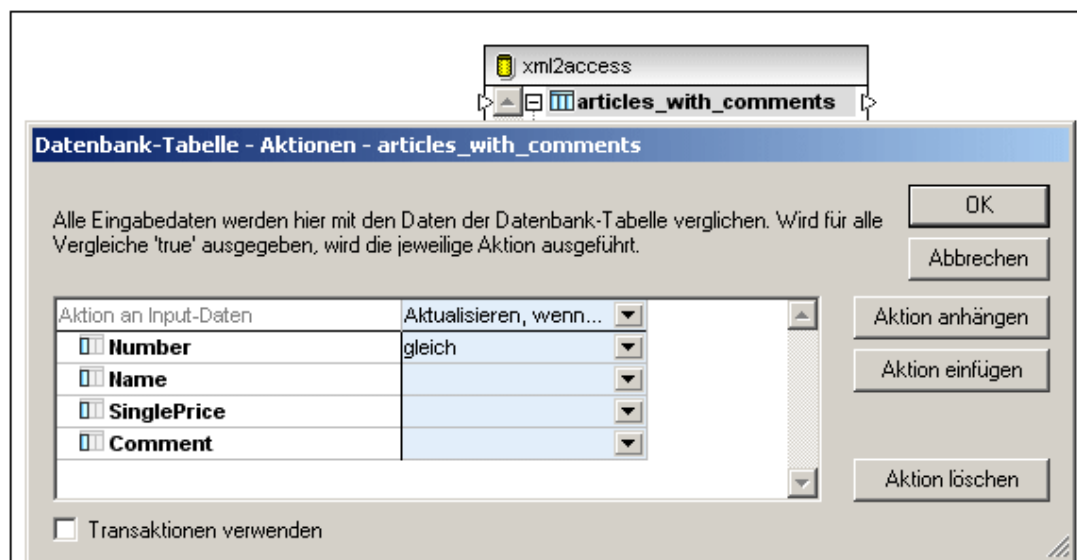


So definieren Sie die Tabellenaktionen und mappen die Daten:

- Klicken Sie auf das Datenbankaktionssymbol **A:In** neben **articles_with_comments**.
- Klicken Sie auf die Auswahlliste **Alles einfügen** und ändern Sie ihn in **Aktualisieren, wenn...**



- Klicken Sie auf die Auswahlliste neben **Number** und wählen Sie **gleich**, klicken Sie anschließend auf OK.



- Klicken Sie auf die Schaltfläche **Ausgabe** und anschließend auf die Schaltfläche **Zeilenumbruch** , um mehr vom SQL Script zu sehen.
- Klicken Sie auf die Schaltfläche "SQL-Skript ausführen" , um die XML-Daten in die Datenbank einzufügen.

```
UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Vernon Callaby"><text>v.callaby@nanonull.com</text></comment></comments>' WHERE ([articles_with_comments].[Number]=1)
-->>> OK. 1 row(s).

UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Frank Further"><text>f.further@nanonull.com</text></comment></comments>' WHERE ([articles_with_comments].[Number]=2)
-->>> OK. 1 row(s).

UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Loby Maise"><text>l.maise@nanonull.com</text></comment></comments>' WHERE ([articles_with_comments].[Number]=3)
-->>> OK. 1 row(s).
```

17.13 IBM DB2 - Mappen von XML-Daten von / auf Datenbanken

MapForce unterstützt derzeit das Mappen von XML-Daten von / auf Datenbanken von Typ IBM DB2 Version 9. In den Beispielen in diesem Abschnitt wird davon ausgegangen, dass Sie Zugriff auf IBM DB 2-Datenbanken haben; alle anderen erforderlichen Dateien finden Sie im Ordner ...**Tutorial**. Ein analoges Mapping-Beispiel für das Mappen von XML-Daten auf eine MS Access-Datenbank, [xml2access.mfd](#), ist zur Gänze verfügbar.

Damit die XML-Nodes in der Datenbankkomponente angezeigt werden, sodass Sie Einträge von und auf die XML-Einträge mappen können, muss ein Schema zugewiesen werden. Anschließend können Sie mit dem Mapping fortfahren, wie gewöhnlich.

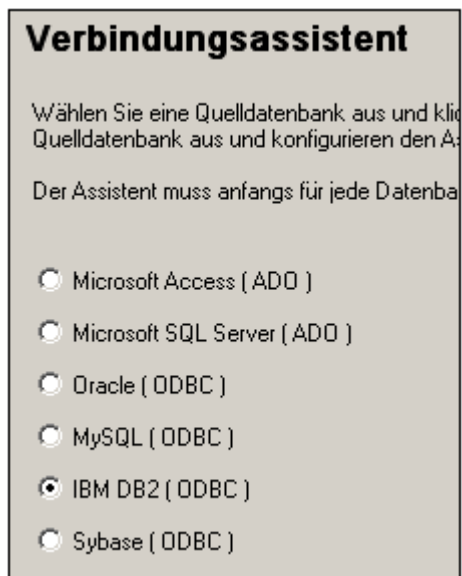
Bitte beachten Sie:

Beim Hinzufügen einer ODBC-Datenquelle für die IBM iSeries (vormals AS/400) wird ein Standard-Flag gesetzt, mit dem Abfrage-Timeouts aktiviert werden. Diese Einstellung muss **deaktiviert** werden, damit Mapping-Dateien in MapForce korrekt geladen werden können.

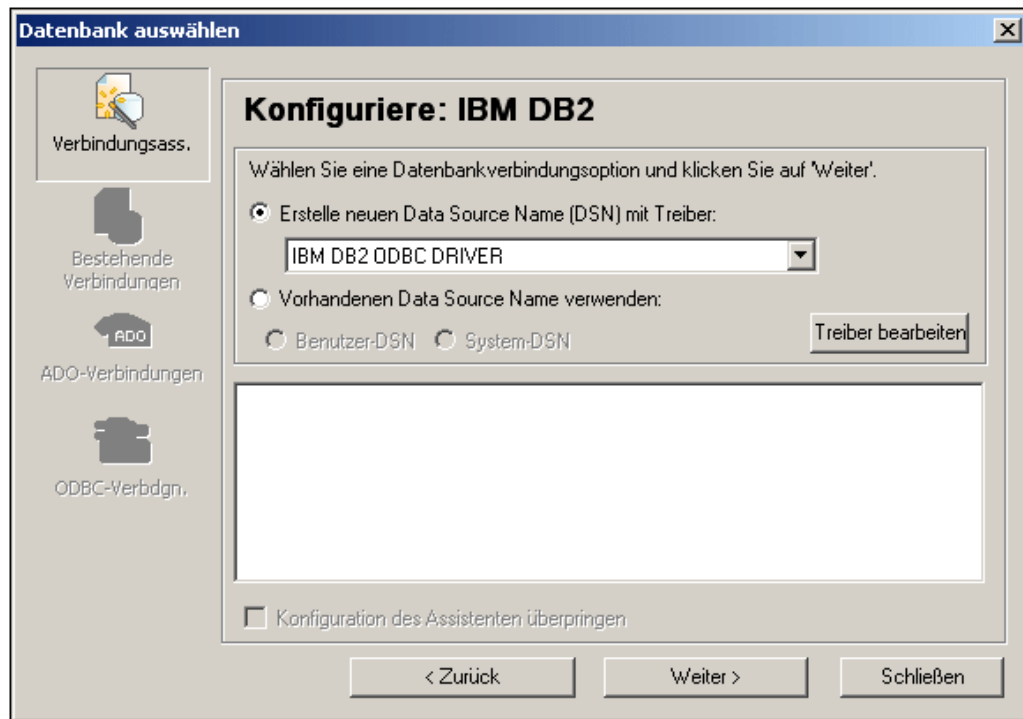
Beim Hinzufügen einer ODBC-Datenquelle für den iSeries Access ODBC-Treiber, wird das Dialogfeld "iSeries Access for Windows ODBC Setup" geöffnet. Wählen Sie das Register "Performance" aus, klicken Sie auf die Schaltfläche "Advanced" und **deaktivieren** Sie das Kontrollkästchen "Allow query timeout".

Konfigurieren und Einfügen einer IBM DB2-Datenbank:

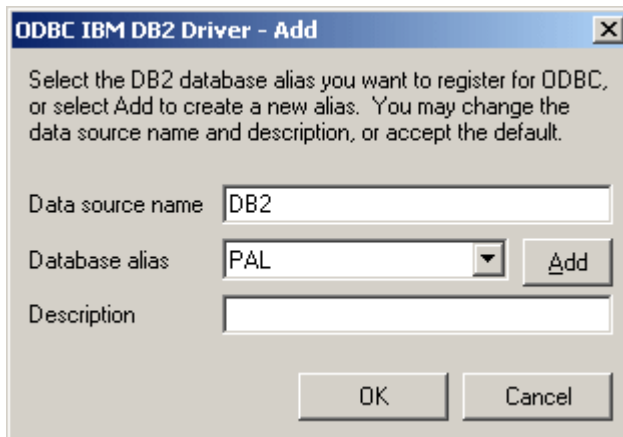
1. Klicken Sie in der Symbolleiste auf eine der Schaltflächen: Java, C#, C++ oder BUILTIN.
2. Klicken Sie in der Symbolleiste auf die Schaltfläche **Datenbank einfügen** .



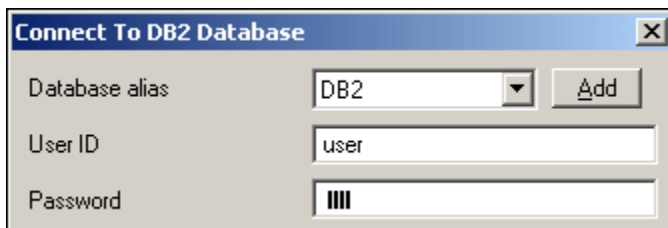
3. Klicken Sie auf das Optionsfeld IBM DB2 (ODBC) und anschließend auf Weiter.
4. Wählen Sie aus der Treiberliste durch Klicken auf den entsprechenden Eintrag einen Treiber aus.



5. Klicken Sie auf **"Weiter"**, um einen neuen DSN (Data Source Name = Datenquellennamen) zu definieren.
6. Geben Sie den Datenquellennamen ein, wählen Sie den Datenbank-Alias aus (oder fügen Sie ihn hinzu) und klicken Sie auf OK.



7. Geben Sie die Anmeldedaten ein und klicken Sie auf OK, um eine Verbindung zur Datenbank herzustellen.

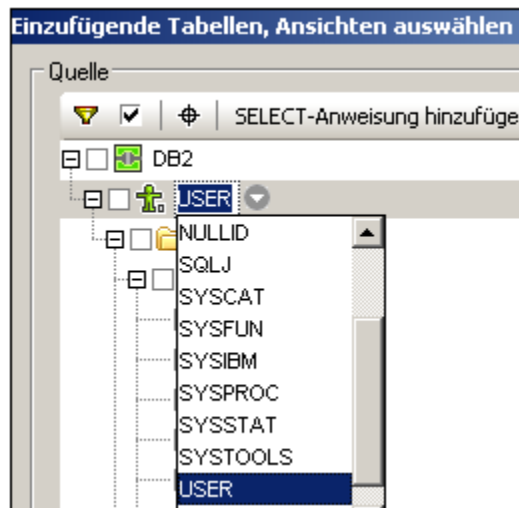


Wenn Sie die Login-Daten eingeben und auf OK klicken, wird ein weiteres Dialogfeld

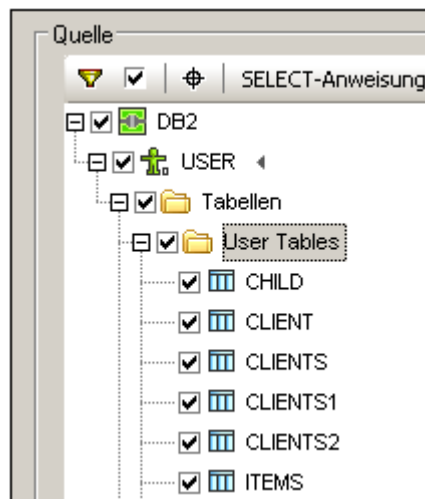
geöffnet, in dem Sie auswählen können, wie die Datenquelle mit dem Server kommunizieren soll, nativ oder über die ODBC API.

Daraufhin wird das Dialogfeld "Datenbankobjekte auswählen" geöffnet.

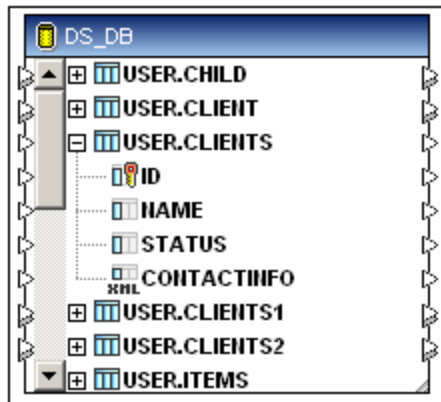
8. Klicken Sie auf die Schaltfläche "Datenbankschema"  und wählen Sie das richtige Datenbankschema aus. z.B. USER.



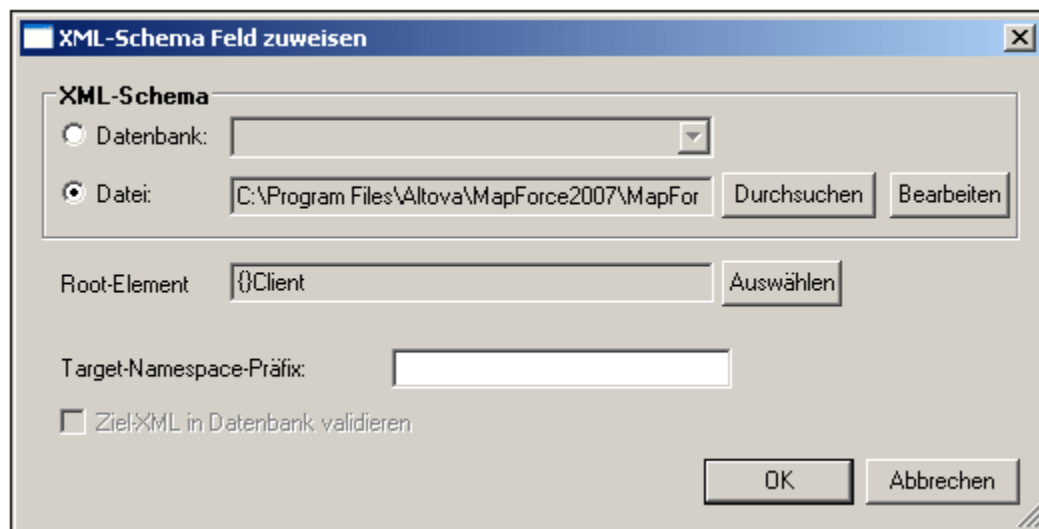
Sie sehen daraufhin alle USER-Tabellen.



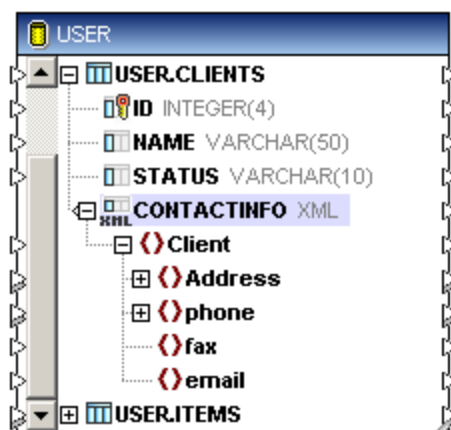
9. Aktivieren Sie das Kontrollkästchen neben "User Tables", um alle untergeordneten Tabellen auszuwählen und klicken Sie auf OK, um die Datenbankkomponente einzufügen.
10. Klicken Sie auf das Erweiterungssymbol neben der Tabelle USER.CLIENTS, um ihren Inhalt zu sehen.
Beachten Sie: die Spalte CONTACTINFO ist vom Typ **XML**.

**Zuweisen eines XML-Schemas zu einer XML-Datei:**

1. Rechtsklicken Sie unter der Tabelle USER.CLIENTS auf die Spalte CONTACTINFO und wählen Sie den Befehl "XML-Schema Feld zuweisen".
2. Wenn Sie auf das Optionsfeld **Datenbank** klicken, können Sie aus Schemas auswählen, die in der Datenbank gespeichert (registriert) wurden. Bei Auswahl von **Datei** können Sie ein lokales Schema auswählen. In diesem Beispiel wurde das XML-Schema **DB2Client.xsd** aus dem Ordner ...\\Tutorial ausgewählt.

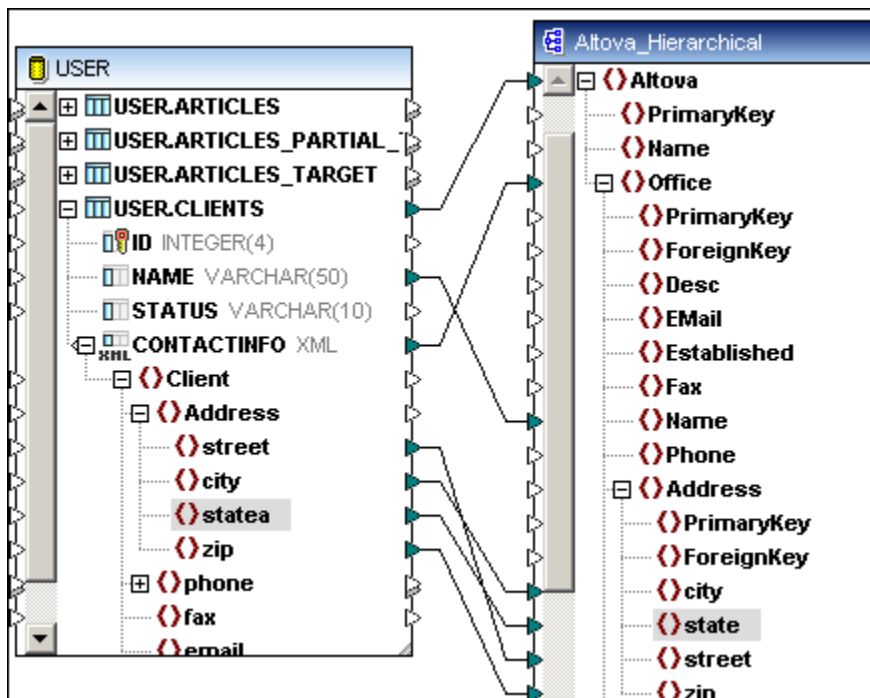


3. Wählen Sie das Root-Element des Schemas, das in der Komponente aufscheinen soll, z.B. Client und bestätigen Sie mit OK.



Das Datenelement "Client" wird unterhalb von CONTACTINFO angezeigt; klicken Sie auf das Erweiterungssymbol, um die Schemastruktur zu sehen.

4. Mappen Sie Konnektoren von den Datenelementen des Schemas zur Zielkomponente, bei der es sich in diesem Fall um ein XML-Dokument handelt.



5. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

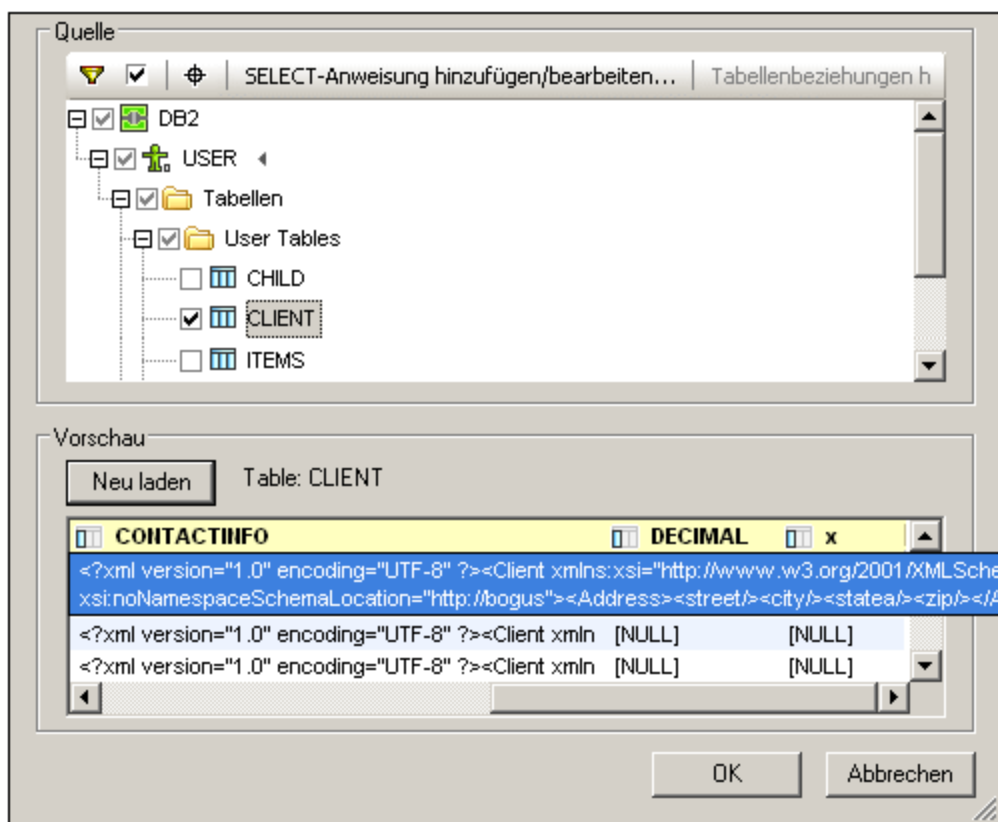
```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Altova xsi:noNamespaceSchemaLocation="C:\PROGRA~
3  <Office>
4      <Name>Ella Kimpton</Name>
5      <Address>
6          <city>San Jose</city>
7          <state>CA</state>
8          <street>5401 Julio Ave.</street>
9          <zip>95116</zip>
10     </Address>
11 </Office>
12 <Office>
13     <Name>Chris Bontempo</Name>

```

Anzeigen einer Vorschau des Tabelleninhalts

Wenn Sie auf die Schaltfläche "Vorschau" klicken, während das Dialogfeld **Tabellen / Ansichten auswählen** im Verbindungsassistenten geöffnet ist, werden die Tabellendaten im Fenster "Vorschau" angezeigt. Wenn Sie auf die Schaltfläche "Vorschau" klicken, ändert sie sich in "Neu laden". Wenn eine Tabellenspalte den Typ "XML" hat, erscheint ein Popup-Fenster mit dem XML-Inhalt, wenn Sie den Mauszeiger über die XML-Spalte platzieren.



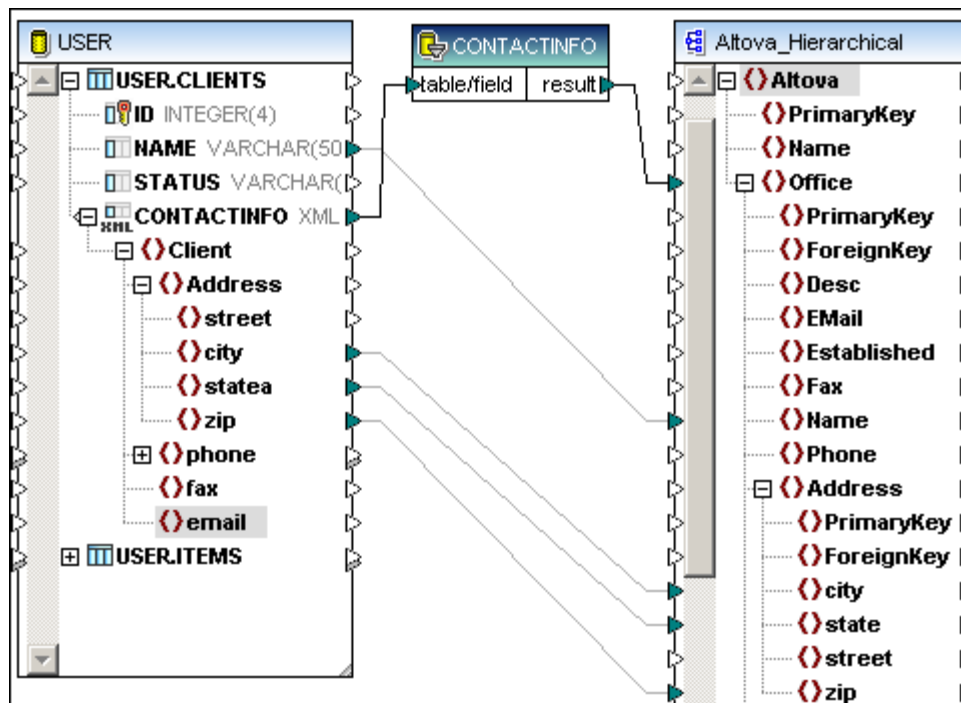
17.13.1 Abfragen und Mappen von XML-Daten in IBM DB2

MapForce gestattet Ihnen, XML-Daten in IBM DB2-Datenbanken mit Hilfe der SQL WHERE-Komponente abzufragen und die Datensätze auf andere Komponenten zu mappen. Nähere Informationen zum Einfügen und Verwenden der SQL WHERE-Komponente finden Sie unter [SQL WHERE-Komponente / Bedingung](#). In diesem Abschnitt wird erläutert, wie man Abfragen an XML-Daten in einer IBM DB2-Datenbank durchführt und die Daten mappt.

Beachten Sie, dass man Datenbanken über das Register "Datenbankabfrage" auch direkt abfragen kann. Nähere Informationen dazu finden Sie unter [Direktes Abfragen von Datenbanken - Register "Datenbankabfrage"](#).

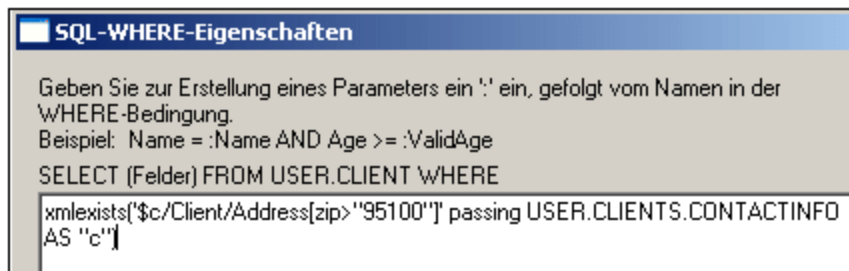
Nachdem Sie die [DB2-Datenbank eingefügt](#) und das XML-Schema dem Datenelement CONTACTINFO zugewiesen haben:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche SQL WHERE , um die Anweisung einzufügen.
2. Verbinden Sie das Datenelement **CONTACTINFO** der Datenbankquelle mit dem Datenelement **table** der Komponente.
3. Verbinden Sie das Datenelement **result** mit einem Datenelement in der Zielkomponente z.B. Office.



Daraufhin wird der Name der SQL WHERE-Komponente auf CONTACTINFO aktualisiert.

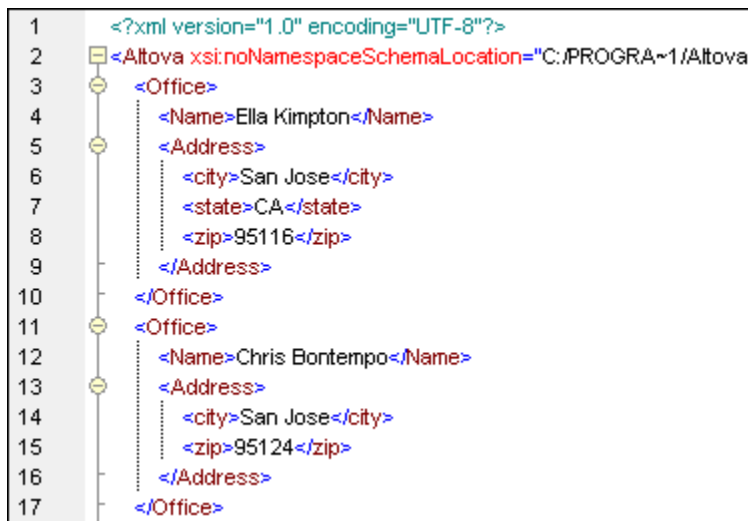
4. Doppelklicken Sie auf die Komponente CONTACTINFO, um die Abfrage zu erstellen.
5. Geben Sie die SQL/XML WHERE Abfrage ein z.B. `xmlexists('$c/Client/Address [zip>"95100"]' passing USER.CLIENTS.CONTACTINFO AS "c")`



Beachten Sie, dass der erste Teil der SELECT-Anweisung, SELECT (fields) FROM USER CLIENTS WHERE, automatisch für Sie generiert wird, wenn Sie den Input- und den Output-Konnektor mit der Datenbank und der Zielkomponente verbinden.

Diese Abfrage gibt diejenigen Datensätze aus, in denen der zip-Code in der XML-Datei größer als 95100 ist.

Mit Hilfe der Funktion **xmlexists** können Sie mittels eines XPath-Ausdrucks zu einem XML-Dokument navigieren, z.B. '`$c/Client/Address[zip>"95100"]`' und eine Bedingung testen. Nähere Informationen zu SQL/XML-Funktionen finden Sie auf der Webseite [DB2 Information Centre](#).



17.13.2 Mappen von XML-Daten - IBM DB2 als Ziel

In diesem Abschnitt wird erläutert, wie XML-Daten auf eine in einer IBM DB2-Datenbank gespeicherte XML-Zielkomponente gemappt werden. In dem Beispiel wird davon ausgegangen, dass Sie Zugriff auf eine IBM DB2-Datenbank haben; alle anderen erforderlichen Dateien liegen im Ordner ...**Tutorial**.

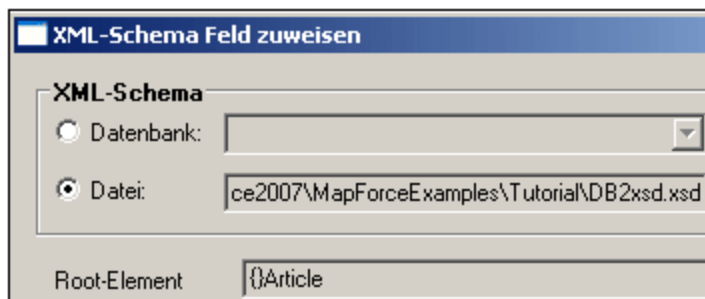
Bitte beachten Sie, dass Sie Datenbanken über das Register "Datenbankabfrage" auch direkt abfragen können. Nähere Informationen dazu finden Sie unter [Direktes Abfragen von Datenbanken - Register "Datenbankabfrage"](#).

So fügen Sie die Datenquellkomponente ein:

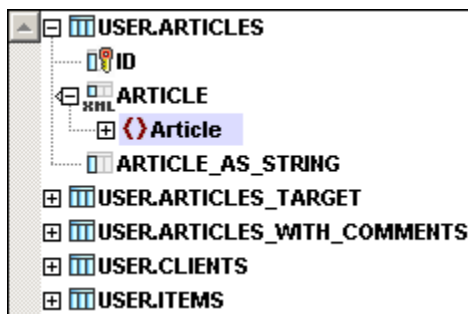
1. Klicken Sie auf die Schaltfläche **XML-Schema/Datei einfügen** , und wählen Sie das Schema **DB2asTarget.xsd** aus.
2. Klicken Sie auf "Durchsuchen...", wenn Sie aufgefordert werden, eine **XML-Beispieldatei** anzugeben und wählen Sie **DB2asTarget.xml** aus.
3. Wählen Sie **Articles** als das Root-Element aus und erweitern Sie es.

So fügen sie das Datenbankziel ein und mappen Daten darauf:

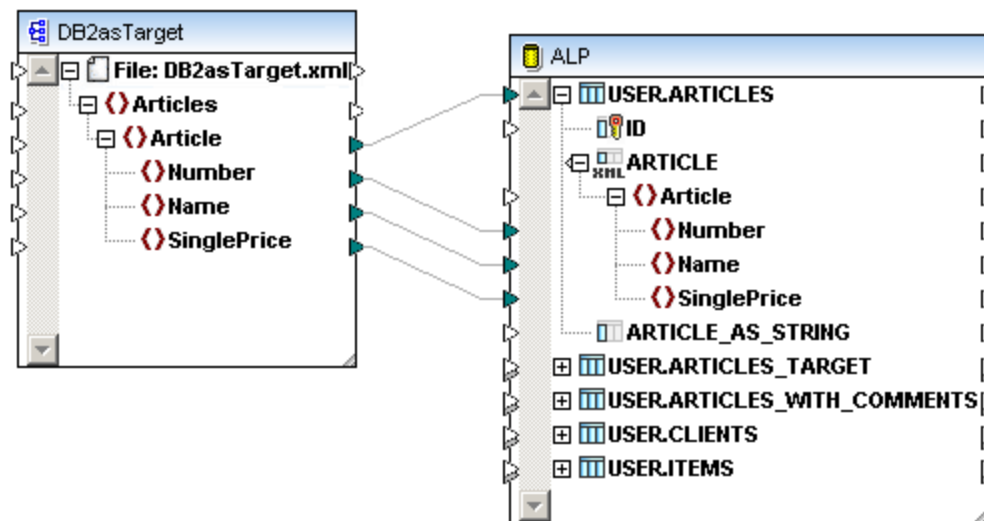
1. Verwenden Sie die unter [Einfügen der IBM DB2-Datenbank](#) beschriebene Methode, um die Datenbank einzufügen.
2. Rechtsklicken Sie auf das Datenelement/die Spalte ARTICLE in der Datenbank-Zielkomponente und wählen Sie den Befehl **XML-Schema Feld zuweisen...**
3. Aktivieren Sie das Optionsfeld **Datei**, wählen Sie **DB2xsd.xsd** und klicken Sie auf OK.



4. Erweitern Sie das Datenelement **Article**, um Konnektoren zu den entsprechenden Datenelementen zu erstellen.



5. Erstellen Sie Verbindungen zwischen der Quell- und der Zielkomponente, wie in der Abbildung unten gezeigt.



6. Klicken Sie auf die Schaltfläche **Ausgabe** und anschließend auf die Schaltfläche

"Zeilenumbruch" , um mehr vom SQL-Skript zu sehen.

```

8      INSERT INTO "USER"."ARTICLES" ("ARTICLE")
9      ... VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
      xsi:noNamespaceSchemaLocation="C:\Program
      Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>11</Number><Name>
      sneakers</Name><SinglePrice>99</SinglePrice></Article>')
10
11      INSERT INTO "USER"."ARTICLES" ("ARTICLE")
12      ... VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
      xsi:noNamespaceSchemaLocation="C:\Program
      Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>22</Number><Name>
      band</Name><SinglePrice>2.9</SinglePrice></Article>')
  
```

Sie sehen nun eine Vorschau der XML-Daten, die in die Datenbank eingefügt werden.

7. Klicken Sie auf die Schaltfläche "SQL-Skript ausführen" , um die Daten einzufügen.

```

1      /*
2      The following SQL statements were executed during "Generate output" function.
3      Every single result is written right to the "-->>>" string.
4      These statements are only for preview and may not be executed in another SQL query tool!
5      */
6
7      INSERT INTO "USER"."ARTICLES" ("ARTICLE")
8      ... VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
9      xsi:noNamespaceSchemaLocation="C:\Program
10     Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>11</Number><Name>s
      sneakers</Name><SinglePrice>99</SinglePrice></Article>')
      -->>> OK. 1 row(s).
  
```

Im Ausgabefenster sehen Sie nun, ob die Befehle erfolgreich ausgeführt wurden.

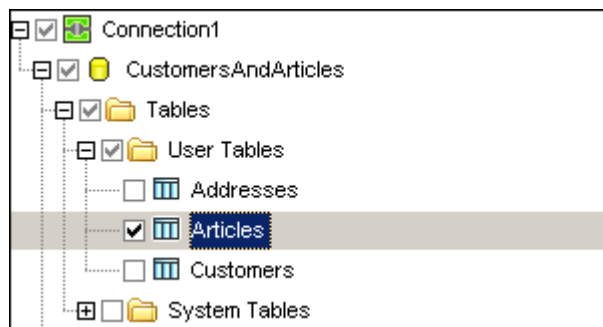
17.13.3 Mappen von Daten - Datenbank auf Datenbank

In diesem Abschnitt wird erläutert, wie XML-Daten von einer MS Access-Datenbank auf ein in einer IBM DB2-Datenbank gespeichertes XML-Zieldokument gemappt werden. In dem Beispiel wird davon ausgegangen, dass Sie Zugriff auf eine IBM DB2-Datenbank haben; alle anderen erforderlichen Dateien liegen im Ordner **...MapForceExamples** oder **...MapforceExampes\Tutorial**.

Bitte beachten Sie, dass Sie Datenbanken über das Register "Datenbankabfrage" auch direkt abfragen können. Nähere Informationen dazu finden Sie unter [Direktes Abfragen von Datenbanken - Register "Datenbankabfrage"](#).

So fügen Sie die MS Access-Datenquellenkomponente ein:

1. Klicken Sie auf die Schaltfläche **Datenbank einfügen**  und wählen Sie die Datenbank **CustomersAndArticles.mdb** aus dem Ordner [...MapForceExamples](#) aus.
2. Wählen Sie die Tabelle **Articles** aus und klicken Sie auf OK, um sie einzufügen.



Die Datenbanktabelle wird nun als Datenbankkomponente angezeigt.



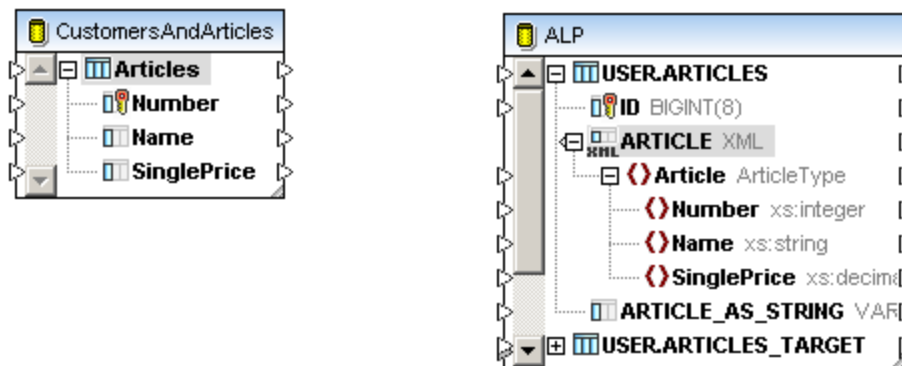
So fügen Sie die IBM DB2 Zieldatenbank ein und mappen Daten darauf:

1. Verwenden Sie die unter [Einfügen der IBM DB2-Datenbank](#) beschriebene Methode, um die Datenbank einzufügen.

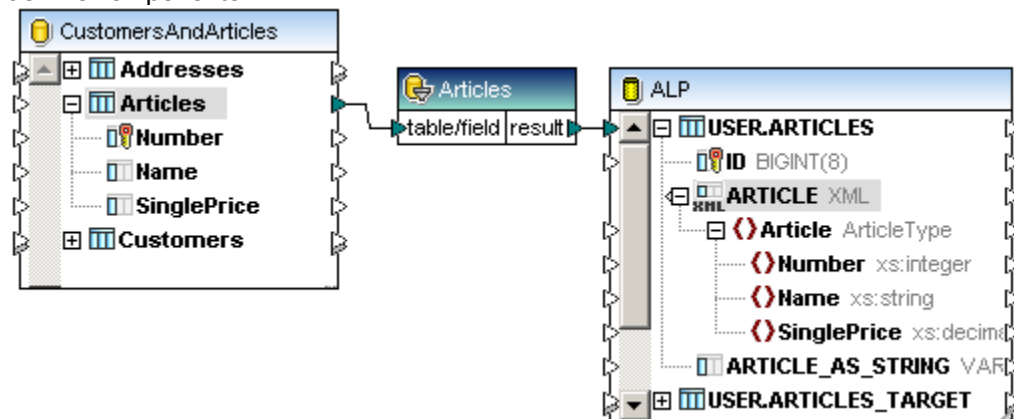


2. Rechtsklicken Sie in der Datenbank-Zielkomponente auf das Datenelement/die Spalte **ARTICLE** und wählen Sie den Befehl **XML-Schema Feld zuweisen...**

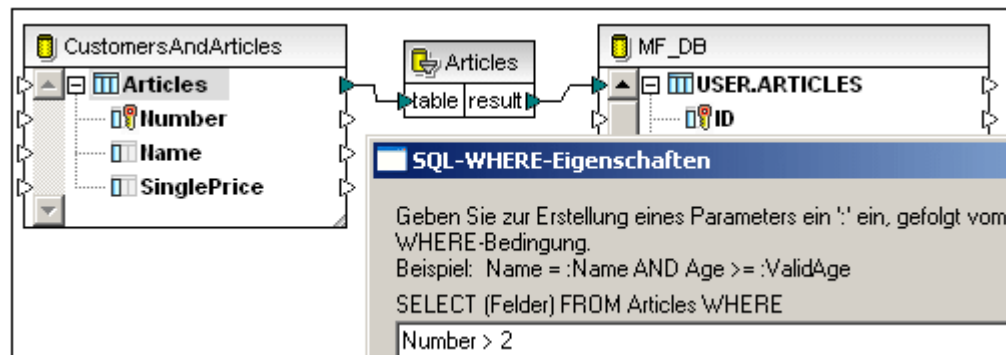
3. Aktivieren Sie das Optionsfeld **Datei**, wählen Sie die Schema-Datei **DB2xsd.xsd** aus und klicken Sie auf OK.



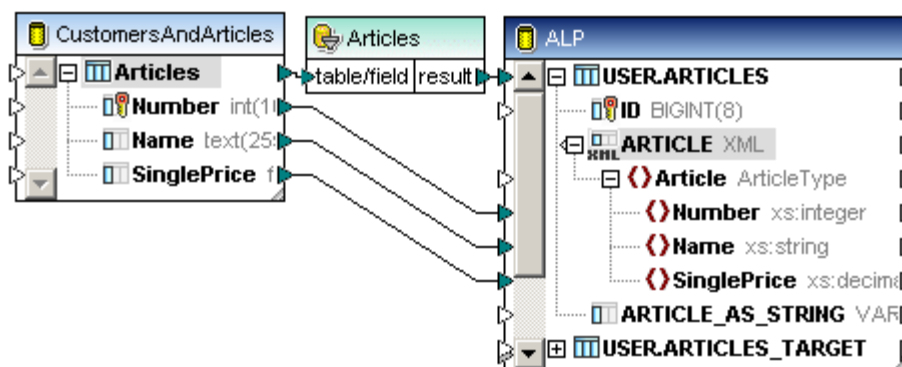
4. Klicken Sie in der Symbolleiste auf die Schaltfläche "SQL WHERE" , um die Komponente einzufügen.
 5. Verbinden Sie das Datenelement **Articles** aus der Datenbankquelle mit der Datenelementkomponente **table**.
 6. Verbinden Sie das Datenelement **result** mit dem Datenelement USER.ARTICLES in der Zielkomponente.



7. Doppelklicken Sie auf die SQL WHERE-Komponente **Articles**, geben Sie als Where-Klausel **Number > 2** ein und klicken Sie auf OK.



8. Mappen Sie die Datenelemente Number, Name und SinglePrice von der Quell- auf die Zieldatenbank.



9. Klicken Sie auf die Schaltfläche "Ausgabe", um eine Vorschau zu sehen. Klicken Sie anschließend auf die Schaltfläche "SQL-Skript ausführen" , um die Daten einzufügen.

```
INSERT INTO "USER"."ARTICLES" ("ARTICLE")
VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
xsi:noNamespaceSchemaLocation="C:\Program
Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>3</Number><Name>Pants<
/Name><SinglePrice>34</SinglePrice></Article>')
```

```
INSERT INTO "USER"."ARTICLES" ("ARTICLE")
VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
xsi:noNamespaceSchemaLocation="C:\Program
Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>4</Number><Name>Jacket
</Name><SinglePrice>5750</SinglePrice></Article>')
```

17.14 SQL Server 2005 - Mappen von XML-Daten

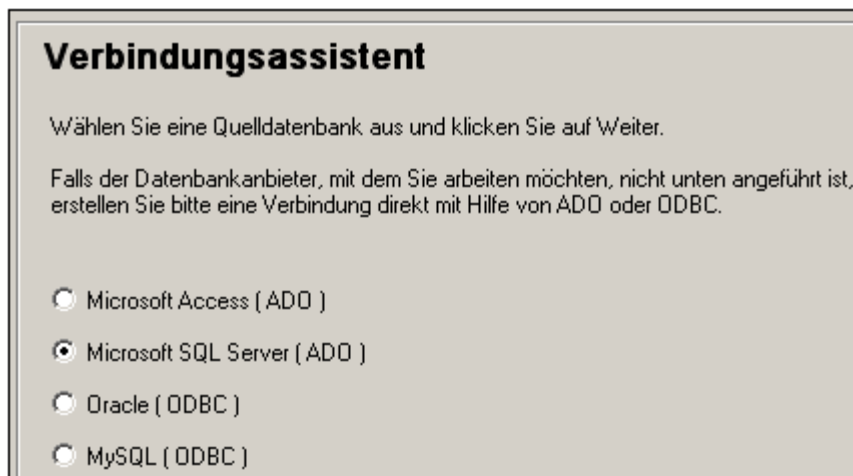
MapForce unterstützt das Mappen von XML-Daten von/auf SQL Server 2005-Datenbanken (und höher). Bei den Beispielen in diesem Abschnitt wird vorausgesetzt, dass Sie Zugriff auf SQL Server 2005-Datenbanken haben.

Damit die XML-Nodes in der Datenbankkomponente, über die XML-Datenelemente gemappt werden können, aufscheinen, muss ein Schema **zugewiesen** werden, bevor Sie mit dem Mapping beginnen können.

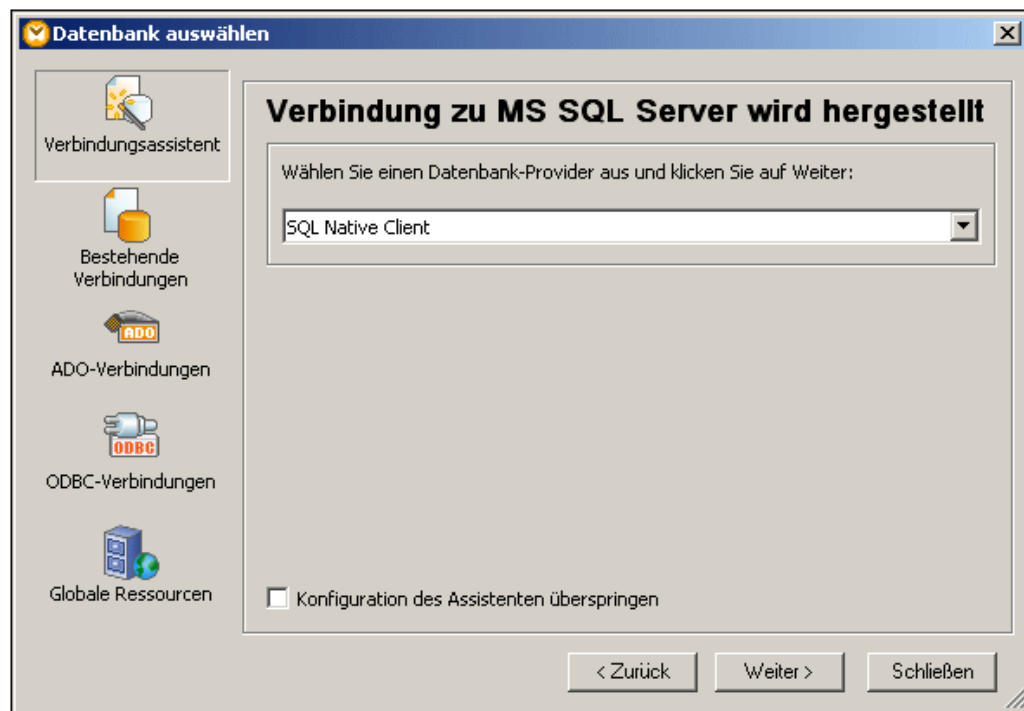
MapForce unterstützt derzeit die Zuweisung **eines** XML-Schemas pro Datenbankspalte und die Auswahl eines **einzigen** "Root"-Elements dieses Schemas.

Konfigurieren und Einfügen einer IBM DB2-Datenbank:

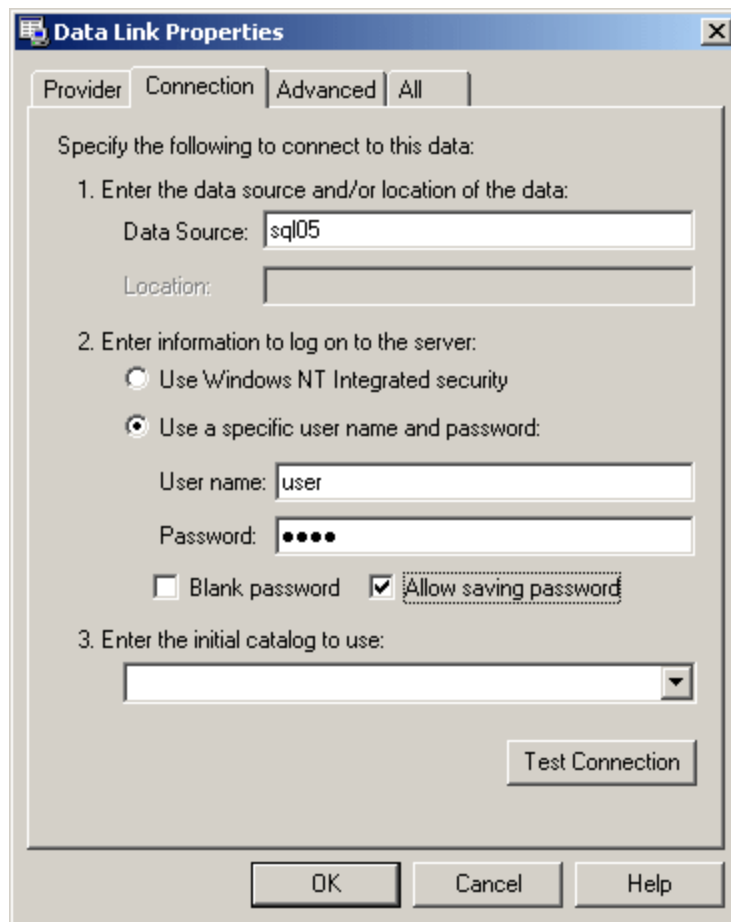
1. Klicken Sie in der Symbolleiste auf eine der Schaltflächen: Java, C#, C++ oder BUILTIN.
2. Klicken Sie in der Symbolleiste auf die Schaltfläche **Datenbank einfügen** .



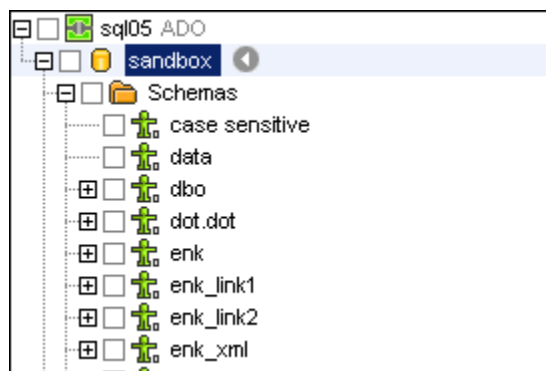
3. Wählen Sie das Optionsfeld "Microsoft SQL Server (ADO)" aus und klicken Sie auf "Weiter".
4. Wählen Sie aus der Treiberliste durch Klicken auf das entsprechende Kontrollkästchen einen Treiber aus.



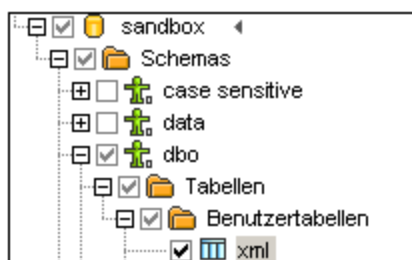
5. Klicken Sie auf **"Weiter"**, um die Data Link-Eigenschaften zu definieren.
6. Geben Sie den Datenquellennamen, den Benutzernamen und das Passwort ein und aktivieren Sie das Kontrollkästchen, mit dem Sie das Speichern des Passwortes gestatten.



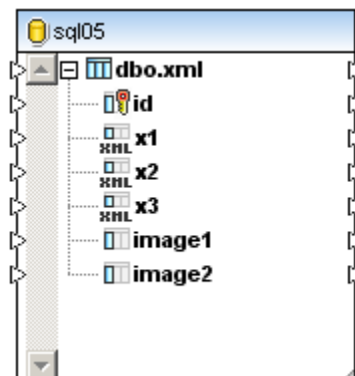
7. Klicken Sie auf die Schaltfläche "Test Connection" und anschließend auf die Schaltfläche "OK", um die Komponente einzufügen.
8. Klicken Sie auf die Schaltfläche "Datenbankschema"  und wählen Sie im Listenfeld das richtige Datenbankschema aus. z.B. sandbox.



9. Aktivieren Sie ein Kontrollkästchen, um die Tabellen, die eingefügt werden sollen, z.B. xml, auszuwählen und klicken Sie auf OK, um die Datenbankkomponente einzufügen. Hinweis: Durch Klicken auf die Schaltfläche "**Vorschau**" können Sie eine Vorschau der Tabelle anzeigen (siehe unten).

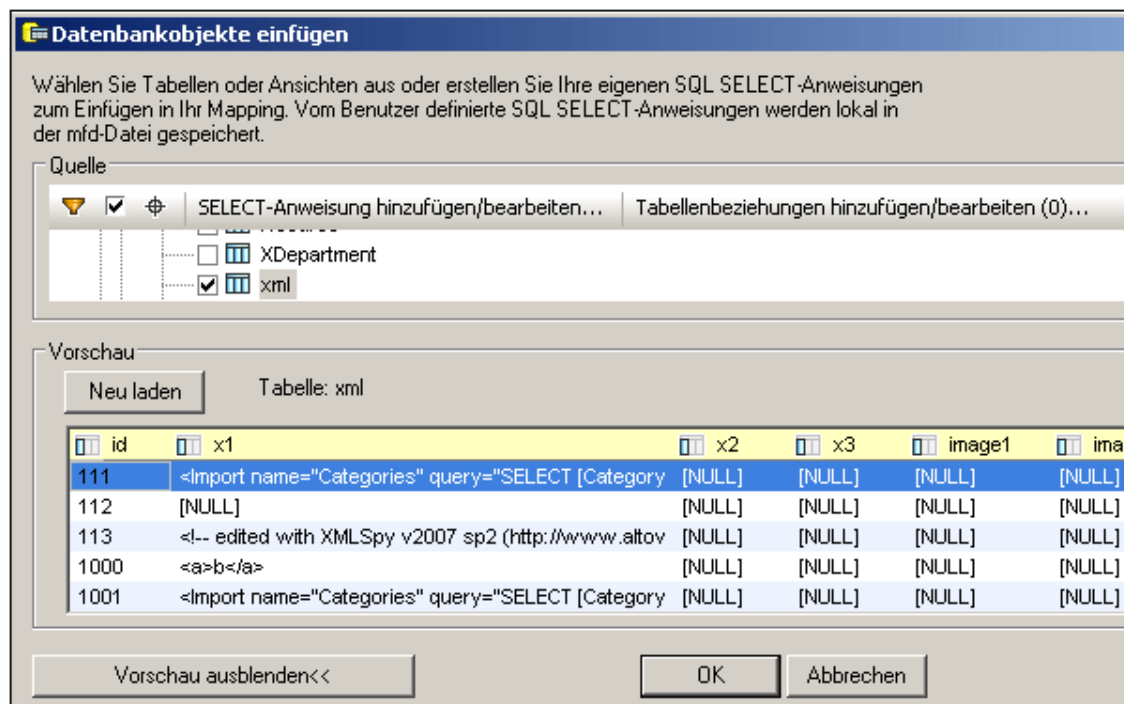


Die XML-Tabelle wurde nun als Datenbankkomponente in MapForce eingefügt.



Beachten Sie, dass die Spalten **x1**, **x2** und **x3** alle vom Typ **XML** sind.

Vorschau der XML-Tabelle:



Zuweisen eines XML-Schemas zu einer XML-Datei:

1. Rechtsklicken Sie auf die Spalte, der Sie das Schema zuweisen möchten, z.B. **x1** und wählen Sie den Befehl **"XML-Schema Feld zuweisen..."**

2. Wenn Sie auf das Optionsfeld **Datenbank** klicken, können Sie aus Schemas auswählen, die in der Datenbank gespeichert (registriert) wurden. Bei Auswahl von **Datei** können Sie ein lokales Schema auswählen.

XML-Schema Feld zuweisen

XML-Schema

☒ Datenbank: dbo.T1

☐ Datei: Durchsuchen Bearbeiten

Root-Element Auswählen

Eingebettete Ziel-XML-Instanz

Target-Namespace-Präfix:

☒ Werte in Zieltypen konvertieren (deaktivieren, um numerische oder Datumswerte beizubehalten, auf die Gefahr hin, dass eine ungültige Ausgabe erzeugt wird)

☐ Ziel-XML in Datenbank validieren

Kodierung

Kodierungsname: Unicode UTF-8

Bytefolge: Little Endian ☐ Bytefolge-Markierung inkludieren

☒ Input-Verarbeitungsoptimierungen auf Basis von min/maxOccurs aktivieren

☒ Alle Dateipfade relativ zur MFD-Datei speichern

OK Abbrechen

3. Klicken Sie auf OK, um die neue Schema-Datei zuzuweisen.

17.15 Direktes Abfragen von Datenbanken - Register "Datenbankabfrage"

MapForce hat ein eigenes Register für Datenbankabfragen, über das Sie Abfragen an jeder gängigen Datenbank durchführen können. Die hier definierte Abfragen/Aktionen sind unabhängig von den anderen MapForce Registern und werden nicht als Teil der *.MFD-Datei gespeichert.

Jedes Datenbankabfrageregister ist mit dem gerade aktiven Mapping verknüpft, sodass mehrere Datenbankabfragen pro Sitzung/Mapping durchgeführt werden können. Beachten Sie, dass Sie für jedes Datenbankabfrage-Register **mehrere aktive Verbindungen** zu unterschiedlichen Datenbanken haben können.

Das Register enthält die folgenden Fenster:

- das **Browser** Fenster auf der linken Seite, in dem die Verbindungsinformationen und die Datenbanktabellen angezeigt werden
- das **SQL Editor**-Fenster rechts vom Browser-Fenster, in dem Sie Ihre SQL-Abfragen verfassen können
- das **Ergebnisfenster**, in dem die Ergebnisse der Abfrage tabellarisch angezeigt werden
- das **Meldungsfenster**, in dem Warn- und Fehlermeldungen angezeigt werden

Die oberste Zeile des Fensters "Datenbankabfrage" enthält die Steuerelemente zum Herstellen der Verbindung mit der Datenbank. Mit Hilfe dieser Schaltflächen können Sie die Arbeitsdatenbanken sowie die Verbindung und die Datenbankschemas definieren.

Altova MapForce - [Neues Design1*]

Menu: Datei, Bearbeiten, Einfügen, Projekt, Komponente, Verbindung, Funktion, Ausgabe, Ansicht, Extras
Fenster, Hilfe

Bibliotheken

- core**
 - boolean result
 - number result
 - string result
- logical functions**
 - equal result
 - equal-or-greater result
 - equal-or-less result
 - greater result
 - less result
 - logical-and result
 - logical-not result
 - logical-or result
 - not-equal result
- math functions**
 - add result
 - ceiling result
 - divide result
 - floor result
 - modulus result
 - multiply result
 - round result
 - subtract result

MF_DB | **USER**

1

```
select * from user.clients
where xmlexists(
'$c/Client/Address[zip>'9505
0']' passing USER
.CLIENTS.CONTACTINFO
AS "c")
```

Amount: 200

	ID	NAME	STATUS	CONTACTINFO
1	3227	Ella Kimpton-Smith	Gold	<?xml version="1.0" encodi
2	8877	Chris Bontempo	Gold	<?xml version="1.0" encodi
3	5681	Paula Lipenski	Standard	<?xml version="1.0" encodi

Abruf fertig Rows: 3, Cols: 4 0.125 sec 16:4

Results Messages

Mapping **Datenbankabfrage** Ausgabe

Neues Design1*

Übersicht Meldungen

Bibliotheken hinzufügen...

Bibliotheken Projekt

Neues Design1: Mapping-Vali

Java ausgewählt

17.15.1 Auswählen / Herstellen einer DB-Verbindung

Um eine Abfrage an einer Datenbank durchführen zu können, müssen Sie zunächst eine Verbindung dazu herstellen. Beachten Sie, dass jedes Datenbankabfragefenster mehrere Verbindungen enthalten kann. Wenn Sie auf die Auswahlliste "Datenquelle" klicken, können Sie zwischen den Datenbanken wechseln und über den SQL Editor Abfragen daran ausführen.

In den Beispielen in diesem Abschnitt wird eine Beispieldatenbank verwendet, die mit IBM DB3 Version 9 geliefert wird.

So stellen Sie eine Verbindung zu einer Datenbank her:

1. Klicken Sie im Hauptfenster auf das Register **Datenbankabfrage**.



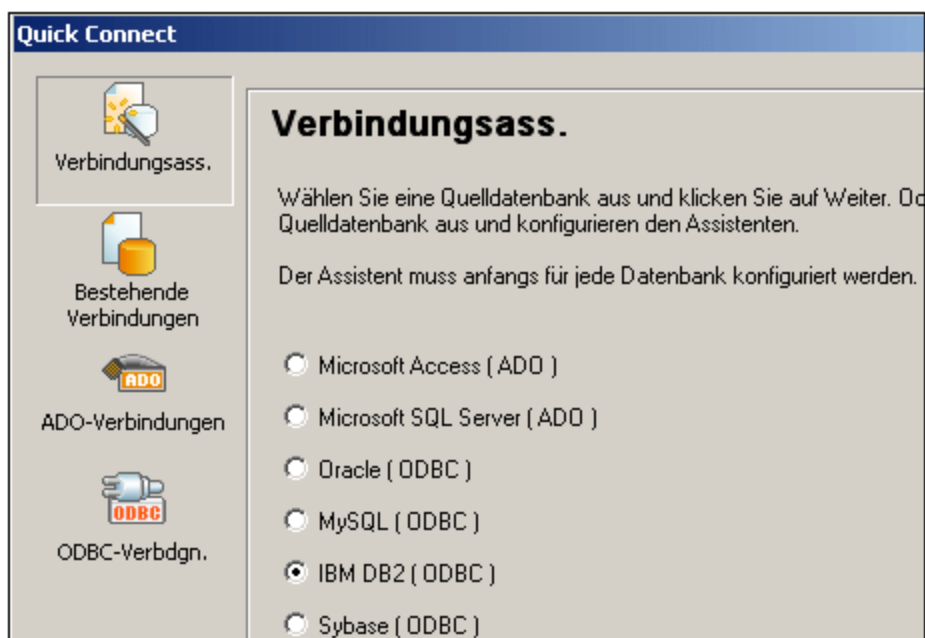
2. Klicken Sie in der Verbindungssymbolleiste auf das Symbol "Verbindungsassistent"



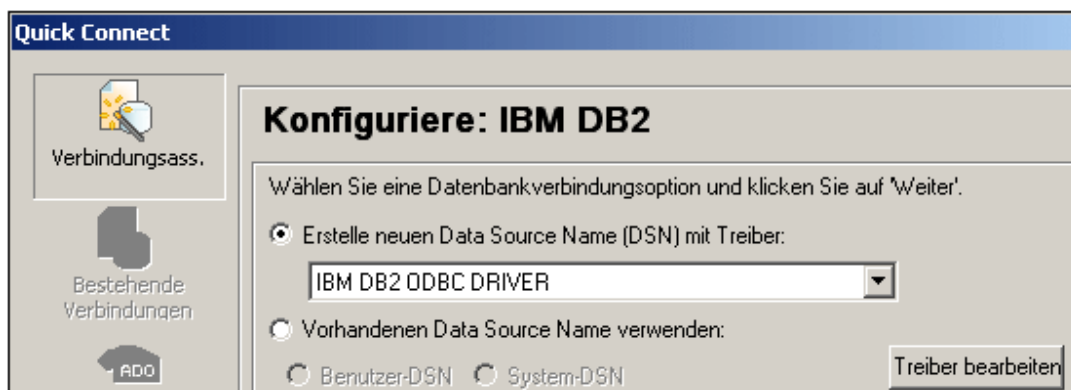
Daraufhin wird der Verbindungsassistent geöffnet, mit Hilfe dessen Sie eine Verbindung zu jeder beliebigen Datenbankart herstellen können. Wenn Sie Mappings geöffnet haben, die Datenbankverbindungen enthalten, werden diese auf der hier gezeigten Seite "Bestehende Verbindungen" angezeigt. In diesem Beispiel gehen wir davon aus, dass derzeit keine Verbindungen aktiv sind.

Wenn Sie im linken Bereich auf das Symbol "Globale Ressourcen"  klicken, können Sie im rechten Bereich eine Datenquelle aus den Datenbanken auswählen, die als globale Ressourcen definiert wurden. Nähere Informationen dazu finden Sie unter [Globale Ressourcen - Datenbanken](#).

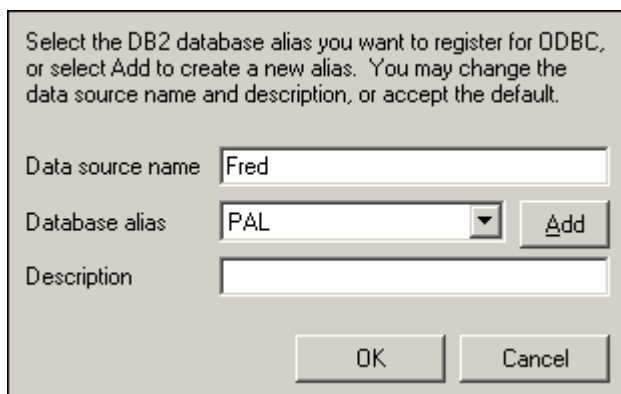
3. Klicken Sie auf das Symbol "Verbindungsassistent", um eine neue Verbindung zu erstellen.



4. Wählen Sie die Datenbank aus, zu der Sie eine Verbindung herstellen möchten, z.B. IBM DB2 und klicken Sie auf Weiter.



5. Klicken Sie auf "Weiter", wenn Sie einen neuen Data Source Name (DSN) erstellen möchten, oder aktivieren Sie das Optionsfeld "Vorhandenen Data Source Name verwenden", wenn Sie zuvor bereits einen DSN definiert haben.



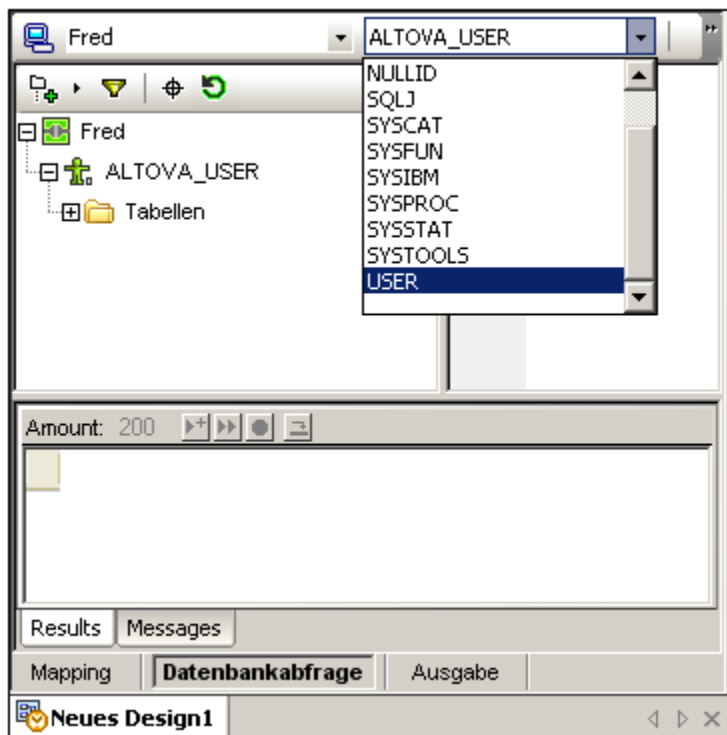
Wenn Sie die Login-Daten eingeben und auf OK klicken, wird ein weiteres Dialogfeld geöffnet, in dem Sie auswählen können, wie die Datenquelle mit dem Server kommunizieren soll, nativ oder über die ODBC API.

6. Geben Sie den DSN sowie den Datenbank-Alias ein und klicken Sie zum Fortfahren auf OK.

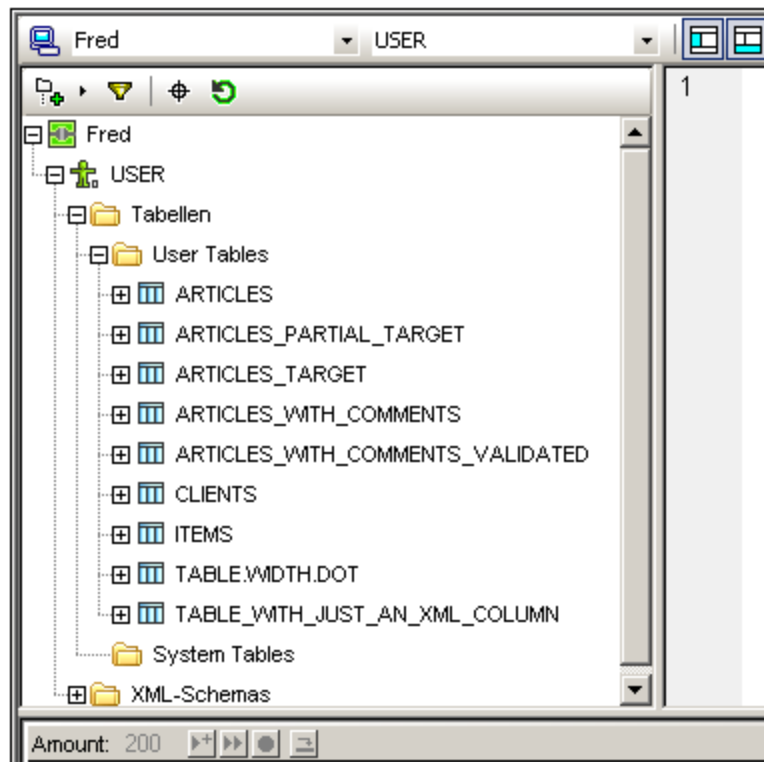


7. Geben Sie die Anmeldedaten für die Datenbank ein und klicken Sie zum Fortfahren auf Weiter.

Es wurde nun eine Verbindung zur DB2-Datenbank hergestellt.



8. Klicken Sie auf die zweite Auswahlliste und wählen Sie das Root-Objekt, z.B. das Datenbank-Schema USER aus.



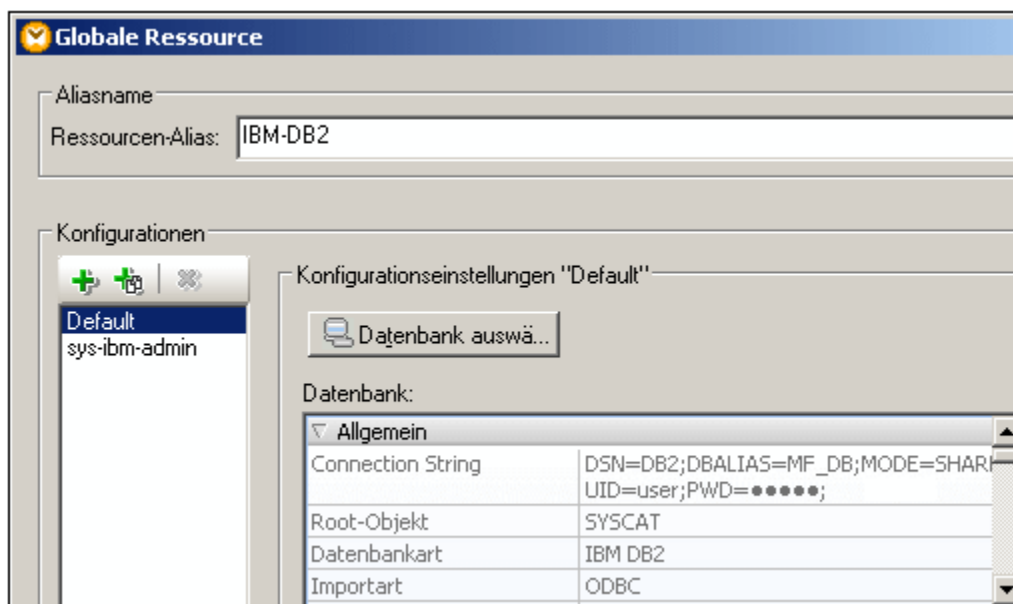
Die Datenbanktabellen werden nun unter dem Ordner "Tabellen" angezeigt. Die Root-Objekte für die verschiedenen Datenbanken sehen Sie in der Tabelle unten:

MS SQL Server	Datenbank
Oracle	Schema
MS Access	Datenbank
MySQL	Datenbank
DB2	Schema
Sybase	Datenbank

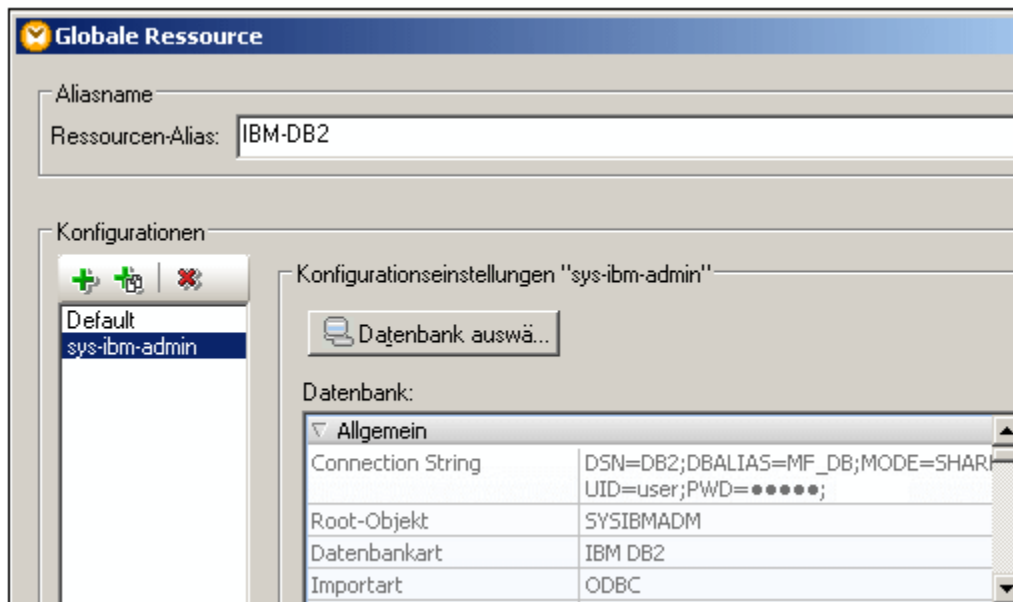
17.15.2 Auswählen einer Datenbank - Globale Ressourcen

Wenn Sie im Dialogfeld "Verbindung zu einer Datenquelle herstellen" auf das Symbol "Globale Ressourcen"  klicken, können Sie eine Datenquelle aus den Datenbanken auswählen, die als globale Ressourcen definiert wurden. Nähere Informationen zum Definieren einer Datenbank als globale Ressourcen finden Sie unter [Globale Ressourcen - Datenbanken](#).

Globale Ressourcen können so definiert werden, dass dieselbe Datenbank verwendet wird, aber für jede Konfiguration andere Tabellen verwendet werden. In der Standardkonfiguration werden die SYSCAT-Tabellen/das Root-Objekt der DB2-Datenbank aufgerufen (siehe Screenshot unten).



Mit der Konfiguration **sys-ibm-admin** dagegen wird dieselbe Datenbank aufgerufen, jedoch nur mit dem SYSIBMADM Tabellen/Root-Objekt.

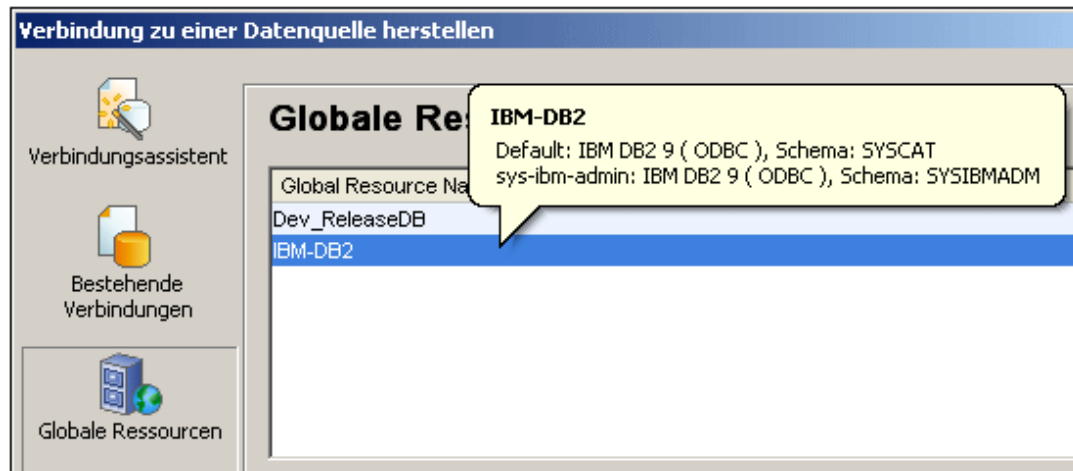


So wechseln Sie die Konfiguration im Fenster DB-Abfrage:

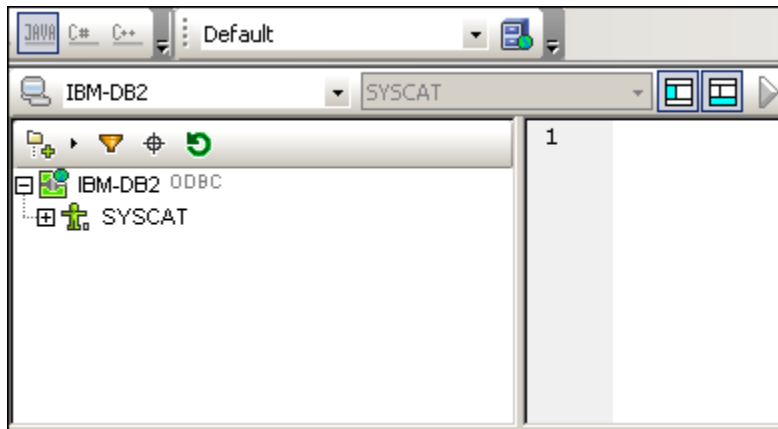
1. Klicken Sie auf das Register "Datenbankabfrage" um das Abfragefenster zu öffnen.
2. Klicken Sie in der Symbolleiste "Verbindung" auf das Symbol "Verbindungsassistent"



und anschließend auf das Symbol "Globale Ressourcen".

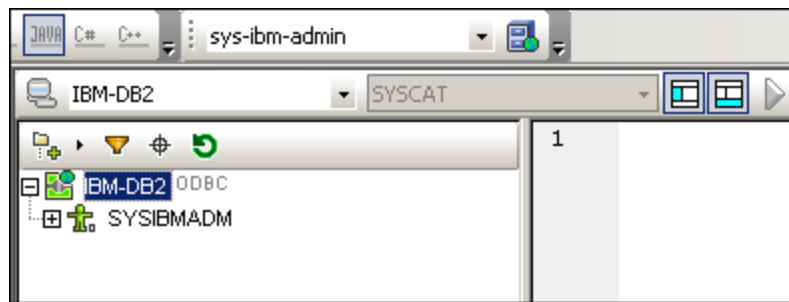


3. Klicken Sie auf den Namen der globalen Ressource, z.B. IBM-DB2 und klicken Sie auf die Schaltfläche "Verbindung herstellen".



Der Name der globalen Ressource ist nun in der linken Auswahlliste sichtbar, wobei das Root-Objekt SYSCAT im rechten Bereich ausgegraut ist.

4. Klicken Sie auf die Auswahlliste der globalen Ressourcen und wählen Sie sys-ibm-admin.
Zu diesem Zeitpunkt wird das Dialogfeld "Konfigurationswechsel - neu laden" geöffnet und Sie werden gefragt, ob die Ressource neu geladen werden soll.
5. Klicken Sie auf die Schaltfläche "Neu laden" und wechseln Sie die Konfiguration.



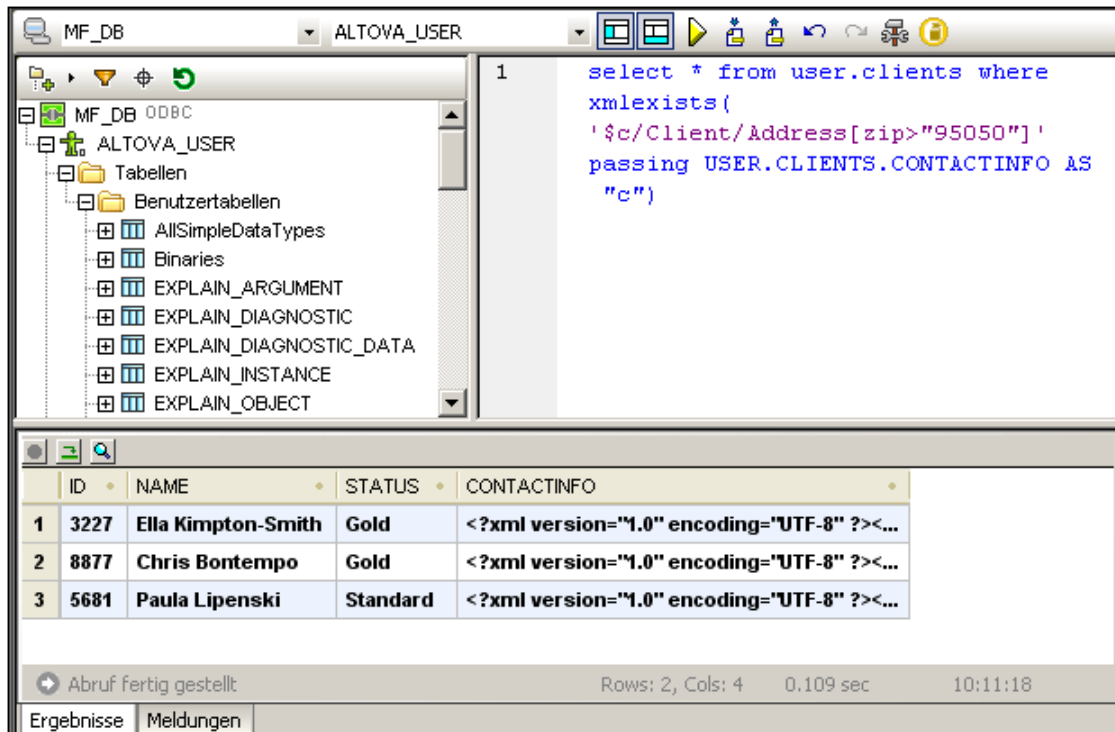
Das neue Objekt / die neue Tabelle ist nun sichtbar und Sie können die einzelnen Tabellen abfragen.

17.15.3 Durchführen von Datenbankabfragen

So rufen Sie Datenbankdaten ab:

Nachdem Sie im vorigen Abschnitt unter [Auswählen / Herstellen einer DB-Verbindung](#) eine Verbindung zur Datenbank hergestellt haben:

1. Klicken Sie auf die Schaltfläche "SQL-Datei importieren"  und wählen Sie im Ordner ...**Tutorial** die Datei **db2-query.sql**. Sie können im Fenster "SQL" auch Ihre eigene SQL-Anweisung eingeben, wenn Sie die mitgelieferte Datei nicht verwenden wollen.
2. Klicken Sie auf die Schaltfläche **Ausführen** .



Die Datenbankdaten werden abgerufen und auf dem Ergebnisregister in tabellarischer Form angezeigt. Beachten Sie, dass in der Statusleiste unter  der aktuelle **Abrufmodus** sowie andere dazugehörige Ergebnisinformationen angezeigt werden.

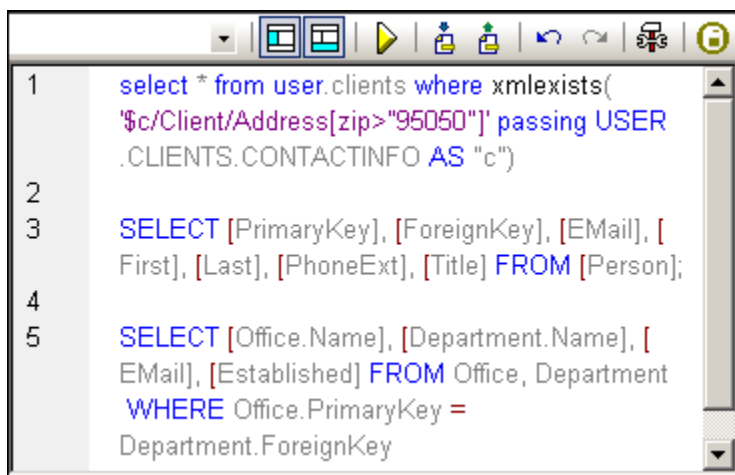
Im Abrufmodus können Sie die Daten auf dem Ergebnisregister kopieren, auswählen oder sortieren, indem Sie auf die rechte Maustaste klicken und die entsprechende Option aus dem Kontextmenü auswählen. Nähere Informationen dazu finden Sie unter [Datenbankabfrage - Ergebnis- und Meldungsregister](#).

17.15.4 Datenbankabfrage - SQL-Fenster

Der **SQL Editor** dient zum Schreiben und Ausführen von SQL-Anweisungen.

SQL Editor-Funktionen:

- **automatischen Generierung** von SQL-Anweisungen mittels **Drag&Drop** aus dem Browser-Fenster
- automatische Komplettierung von SQL-Anweisungen beim Erstellen von SELECT-Anweisungen
- Definition von **Regionen**
- Einfügen von Zeilen- oder Block-Kommentaren



Die SQL-Symbolleiste enthält die folgenden Schaltflächen:



Browser ein/aus: Blendet das Browser-Fenster ein und aus.



Ergebnisse ein/aus: Blendet das Ergebnisfenster ein und aus.



Ausführen (F5): Wenn Sie auf diese Schaltfläche klicken, werden die gerade ausgewählten SQL-Anweisungen ausgeführt. Wenn es mehrere Anweisungen gibt und keine ausgewählt ist, werden alle ausgeführt.



Rückgängig: Dient zum Rückgängigmachen einer unbegrenzten Anzahl von Bearbeitungen im SQL-Fenster.



Wiederherstellen: Dient zum Wiederherstellen von zuvor rückgängig gemachten Befehlen. Mit Hilfe dieser beiden Befehle können Sie sich im Verlauf vor und zurückbewegen.



SQL-Datei importieren: Öffnet eine SQL-Datei im SQL-Editor, die dann ausgeführt werden kann.



SQL-Datei exportieren: Speichert SQL-Abfragen zur späteren Wiederverwendung



SQL Script in DatabaseSpy öffnen: Startet DatabaseSpy und öffnet das Script im SQL Editor-Fenster.



Optionen: Öffnet das Dialogfeld "Optionen", in dem Sie sowohl allgemeine als auch SQL-Editor-Einstellungen definieren können.

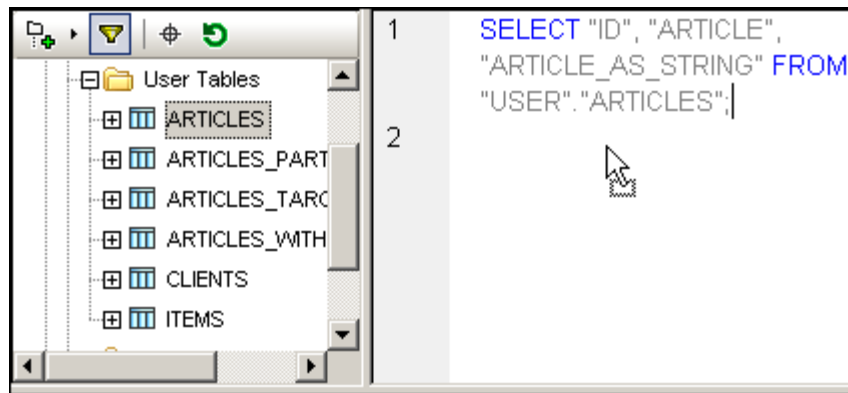
Generieren von SQL-Anweisungen

SQL-Anweisungen, die auf im Browser vorhandenen Tabellen und Spalten basieren, können auf verschiedene Arten automatisch generiert werden.

- durch Ziehen eines Datenbankobjekts aus dem Browser-Fenster in den SQL Editor
- durch Rechtsklick auf ein Datenbankobjekt im Browser und Auswahl der entsprechenden Option aus dem Kontextmenü.

So generieren Sie mit Hilfe von Drag and Drop SQL-Anweisungen:

1. Klicken Sie im Browser auf die Tabelle ARTICLES und ziehen Sie sie in das SQL Editor-Fenster.



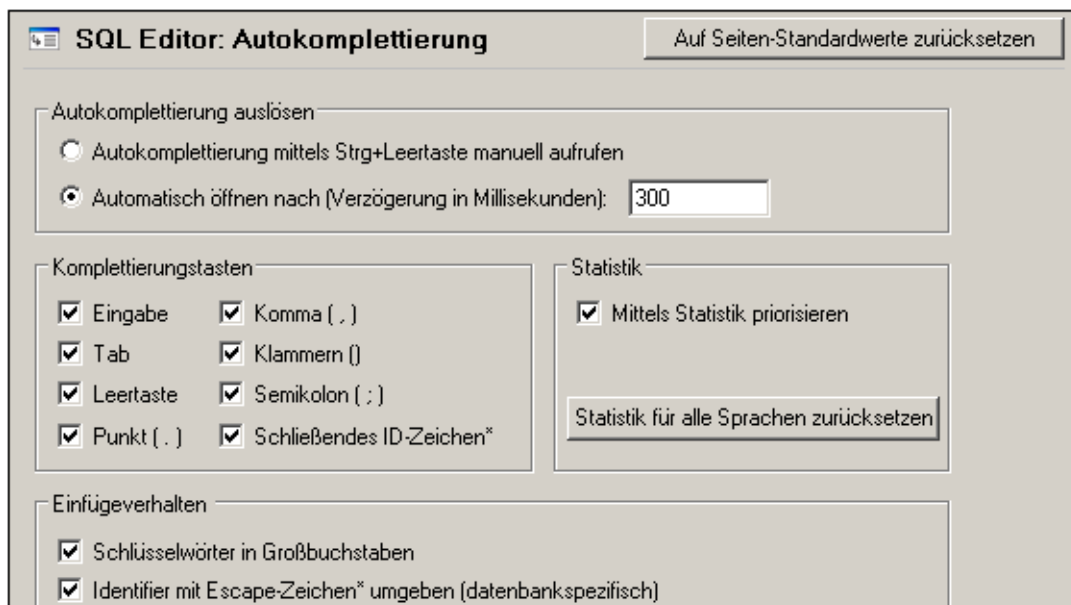
Daraufhin erscheint eine SQL-Anweisung im SQL Editor.

So generieren Sie SQL-Anweisungen über das Kontextmenü:

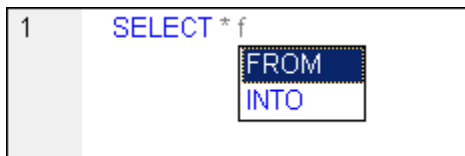
1. Rechtsklicken Sie im Browser auf ein Datenbankobjekt und wählen Sie den Befehl **In SQL Editor anzeigen | Auswählen**.

So erstellen Sie SQL-Anweisungen manuell mit Hilfe der Autokomplettierungsfunktion:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche "Optionen"  und anschließend in der hierarchischen Struktur auf der linken Seite auf den Eintrag "SQL Editor | Autokomplettierung".
2. Wählen Sie die gewünschten Autokomplettierungseinstellungen aus.



3. Beginnen Sie mit der Eingabe der SQL-Anweisung im SQL Editor.



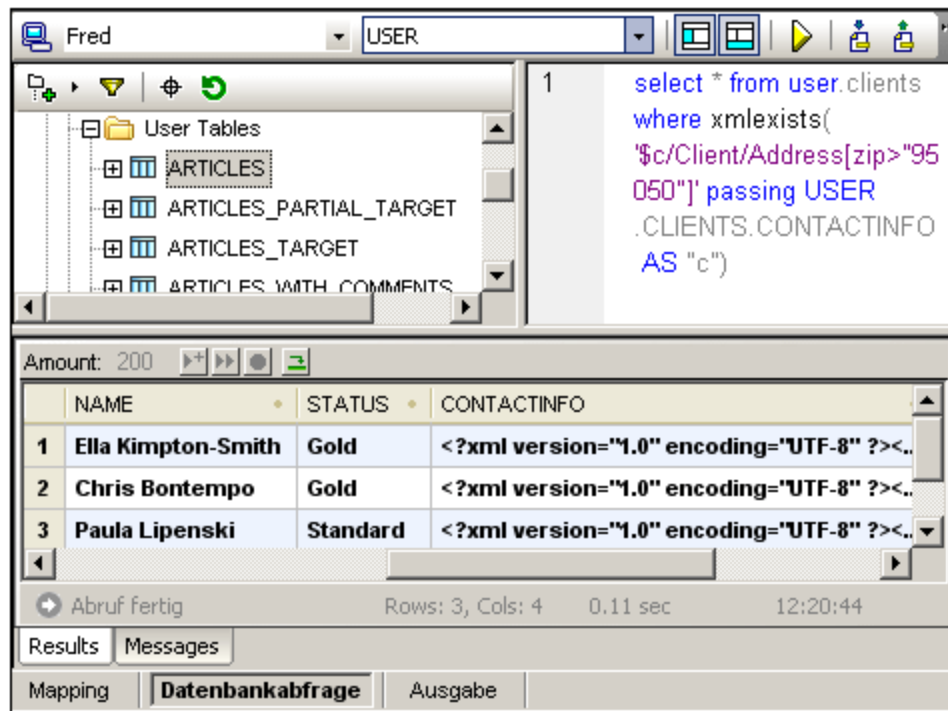
Während der Eingabe der Select-Anweisung werden Autokomplettierungs-Popups angezeigt. Drücken Sie die Eingabetaste, wenn Sie eine markierte Option einfügen/verwenden möchten.

Ausführen von SQL-Anweisungen

SQL-Anweisungen, die erstellt oder geöffnet wurden, können direkt vom SQL Editor aus ausgeführt werden.

So führen Sie eine SQL-Anweisung im SQL Editor-Fenster aus:

1. Geben Sie eine SQL-Anweisung im SQL Editor ein oder wählen Sie eine aus.
2. Klicken Sie auf die Schaltfläche **Ausführen** .



Wenn zu einer [Datenquelle](#) keine Verbindung besteht, erscheint eine Popup-Meldung, in der Sie gefragt werden, ob eine Verbindung hergestellt werden soll.

3. Klicken Sie im Meldungsfeld auf **Ja**, um eine Verbindung zur Datenquelle herzustellen.

So wählen Sie einzelne Anweisungen aus:

- Markieren Sie die jeweilige Anweisung mit Hilfe der Maus.
- Platzieren Sie den Mauszeiger im SQL Editor in die Spalte **Zeilennummer** und klicken Sie, um eine komplette Anweisung auszuwählen.
- Klicken Sie drei Mal in eine ausgewählte Anweisung.

Speichern und Öffnen von SQL Scripts

Sie können jede SQL-Anweisung, die in einem SQL Editor-Fenster angezeigt wird, speichern und das Script später wiederverwenden.

So speichern Sie den Inhalt eines SQL Editor-Fensters in einer Datei:

1. Klicken Sie auf die Schaltfläche **SQL-Datei exportieren**  und geben Sie einen Namen für das SQL Script ein.

So öffnen Sie eine zuvor gespeicherte SQL-Datei:

1. Klicken Sie auf die Schaltfläche **SQL-Datei importieren**  und wählen Sie die gewünschte SQL-Datei im SQL-Fenster aus.

SQL Editor-Funktionen

Sie können SQL-Anweisungen im SQL Editor wie in jedem anderen Text-Editor bearbeiten. Im SQL Editor stehen zusätzliche weitere Funktionen zur Verfügung, wie z.B.:

- [Autokomplettierung](#)
- [Auskommentieren von Text](#)

- [Lesezeichen](#)
- [Regionen](#)

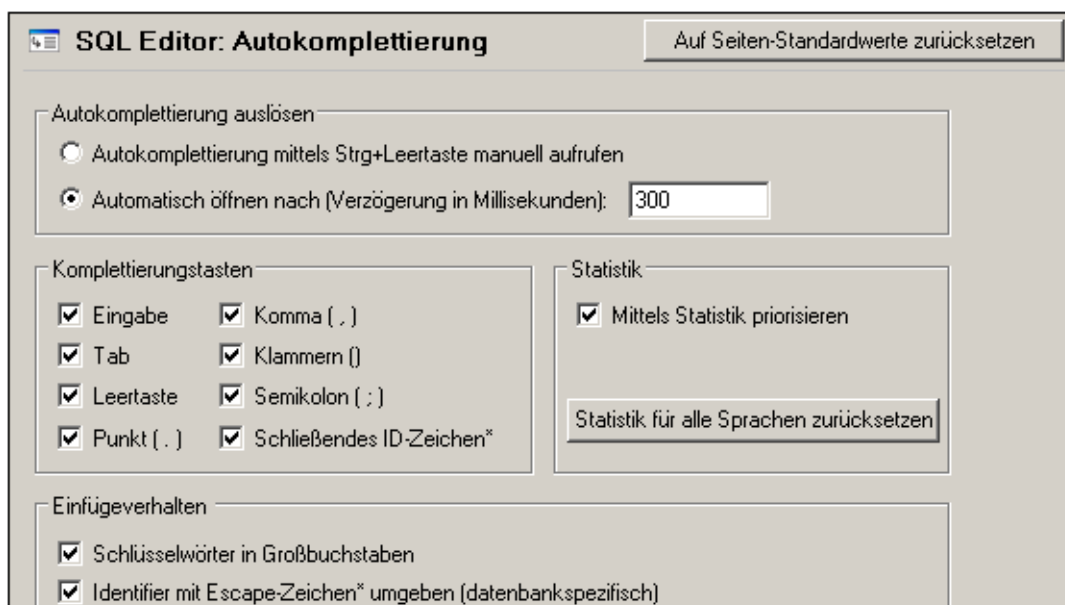
Autokomplettierung

Bei der Eingabe einer SQL-Anweisung im SQL Editor steht Ihnen als Hilfsmittel die Autokomplettierungsfunktion zur Verfügung. Sie können hierbei je nach Art der Anweisung aus Listen der entsprechenden Schlüsselwörter, Datentypen, Identifiern, Trennzeichen und Operatoren auswählen. Die Autokomplettierungsfunktion steht derzeit für die folgenden Datenbanken zur Verfügung:

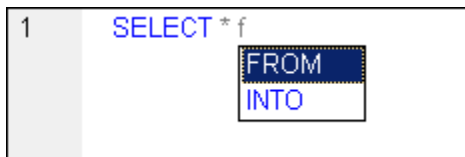
- MS SQL Server 2000
- MS SQL Server 2005
- MS Access 2003
- IBM DB2 9

So aktivieren Sie die Autokomplettierung:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche "Optionen"  und anschließend in der Baumstruktur auf der linken Seite auf den Eintrag **SQL Editor | Autokomplettierung**.
2. Definieren Sie die gewünschten Autokomplettierungseinstellungen.



3. Beginnen Sie mit der Eingabe der SQL-Anweisung im SQL Editor.



Während der Eingabe der Anweisung wird ein Autokomplettierungs-Popup-Fenster angezeigt. Drücken Sie die Eingabetaste, wenn Sie eine markierte Option einfügen möchten.

Auskommentieren von Text

Sie können im SQL Editor Anweisungen, Teile von Anweisungen oder Gruppen von Anweisungen auskommentieren. Diese Anweisungen bzw. die entsprechenden Teile davon werden bei der Ausführung des SQL Scripts übersprungen.

So kommentieren Sie einen Textabschnitt aus:

1. Wählen Sie eine Anweisung oder den Teil einer Anweisung aus.

```
1  select * from user.clients where xmlexists(  
    '$c/Client/Address[zip>"95050"]' passing USER  
    .CLIENTS.CONTACTINFO AS "c")  
2  
3  SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU", "SRP",  
    "COMMENTS" FROM "USER"."ITEMS";  
4  
5  SELECT [PrimaryKey], [ForeignKey], [Name] FROM [  
    Department];
```

2. Rechtsklicken Sie auf die ausgewählte Anweisung und wählen Sie den Befehl **Einfügen / Blockkommentar entfernen**.

```
1  select * from user.clients where xmlexists(  
    '$c/Client/Address[zip>"95050"]' passing USER  
    .CLIENTS.CONTACTINFO AS "c")  
2  
3  /*SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",  
    "SRP", "COMMENTS" FROM "USER"."ITEMS";*/  
4  
5  SELECT [PrimaryKey], [ForeignKey], [Name] FROM [  
    Department];
```

Die Anweisung wird daraufhin auskommentiert.

So kommentieren Sie Text Zeile für Zeile aus:

1. Rechtsklicken Sie auf die gewünschte Stelle, an der Sie den Text auskommentieren möchten und wählen Sie den Befehl **Einfügen / Zeilenkommentar entfernen**.

```
3  SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU", "SRP",  
    "COMMENTS" FROM "USER"."ITEMS";  
4  
5  SELECT [PrimaryKey], --[ForeignKey], [Name] FROM  
    [Department];  
6
```

Die Anweisung wird ab der aktuellen Cursorposition bis zum Ende der Anweisung auskommentiert.

So entfernen Sie einen Block- oder Zeilenkommentar:

1. Wählen Sie den auskommentierten Teil der Anweisung aus.
Um einen Zeilenkommentar zu entfernen, genügt es, nur die Kommentarmarkierungen -- vor dem Kommentar auszuwählen.

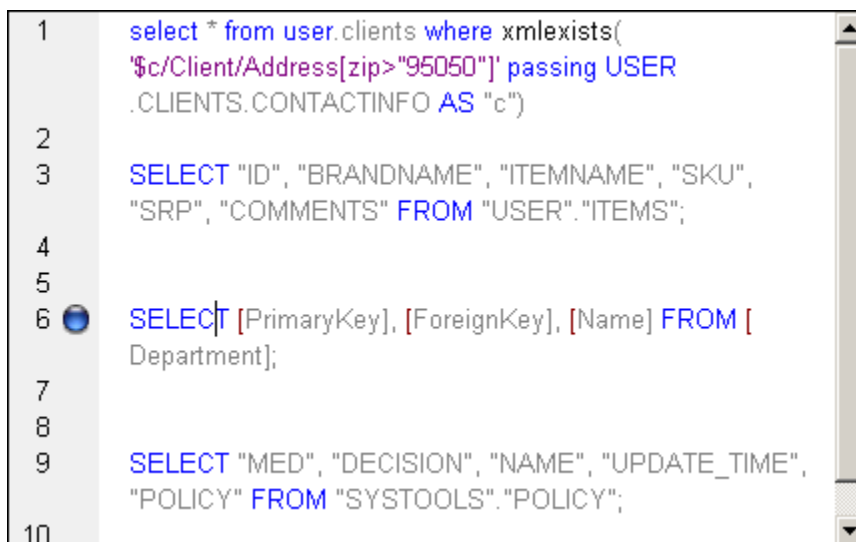
2. Klicken Sie auf die rechte Maustaste und wählen Sie den Befehl **Einfügen / Block-(oder Zeilen) Kommentar entfernen**.

Verwenden von Lesezeichen

Lesezeichen dienen zum Markieren von Elementen in langen Scripts.

So fügen Sie ein Lesezeichen ein:

1. Rechtsklicken Sie auf die Zeile, die Sie mit einem Lesezeichen versehen wollen, und wählen Sie im Kontextmenü den Befehl **Lesezeichen einfügen/entfernen**.



```

1  select * from user.clients where xmlexists(
2    '$c/Client/Address[zip>"95050"]' passing USER
3    .CLIENTS.CONTACTINFO AS "c")
4
5  SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",
6  "SRP", "COMMENTS" FROM "USER"."ITEMS";
7
8  SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
9    Department];
10
11 SELECT "MED", "DECISION", "NAME", "UPDATE_TIME",
12 "POLICY" FROM "SYSTOOLS"."POLICY";

```

Daraufhin wird am Rand am Anfang der markierten Zeile ein Lesezeichensymbol  angezeigt.

So entfernen Sie ein Lesezeichen:

1. Rechtsklicken Sie in die Zeile, aus der Sie das Lesezeichen entfernen möchten und wählen Sie im Kontextmenü den Befehl **Lesezeichen einfügen/entfernen**.

So navigieren Sie zwischen Lesezeichen:

- Um mit dem Cursor zum nächsten Lesezeichen zu gehen, rechtsklicken Sie und wählen Sie den Befehl **Zum nächsten Lesezeichen**.
- Um den Cursor zum vorhergehenden Lesezeichen zu verschieben, rechtsklicken Sie und wählen Sie den Befehl **Zum vorhergehenden Lesezeichen**.

So entfernen Sie alle Lesezeichen

- Rechtsklicken Sie und wählen Sie den Befehl **Alle Lesezeichen entfernen**.

Einfügen von Regionen

Regionen sind Textabschnitte, die zur Strukturierung Ihrer SQL Scripts markiert und als Einheit definiert werden. Regionen können reduziert und erweitert werden, um Teile von SQL Scripts ein- und auszublenden. Regionen können auch innerhalb anderer Regionen verschachtelt werden.

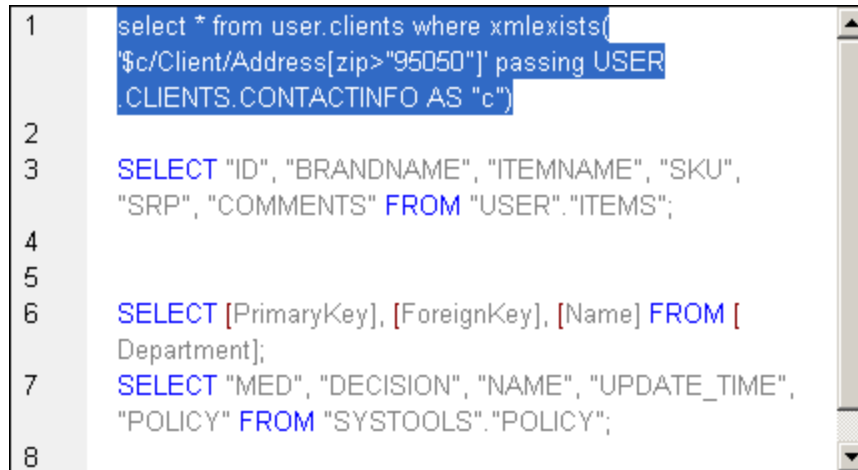
Wenn Sie eine Region einfügen, werden ein Erweitern/Reduzieren Symbol und ein --region Kommentar oberhalb des ausgewählten Texts eingefügt.

Bitte beachten Sie: Sie können den Namen einer Region ändern, indem Sie an den --region Kommentar einen beschreibenden Text anhängen. Das Wort "region" darf nicht gelöscht

werden, z.B. --region DB2query.

So erstellen Sie eine Region:

1. Wählen Sie im SQL Editor die Anweisungen aus, die als Region definiert werden sollen.



```
1 select * from user.clients where xmlexists(  
2     '$c/Client/Address[zip>"95050"]' passing USER  
3     .CLIENTS.CONTACTINFO AS "c")  
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  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

2. Rechtsklicken Sie und wählen Sie im Kontextmenü den Befehl **Region hinzufügen**. Der markierte Bereich wird zu einer Region und kann erweitert und reduziert werden.



```
1 -- region  
2 select * from user.clients where xmlexists(  
3     '$c/Client/Address[zip>"95050"]' passing USER  
4     .CLIENTS.CONTACTINFO AS "c")  
5 -- endregion  
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  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

3. Klicken Sie auf das + oder - Feld, um die Region zu erweitern bzw. zu reduzieren.

So entfernen Sie eine Region:

- Löschen Sie die Kommentare -- region und -- endregion.

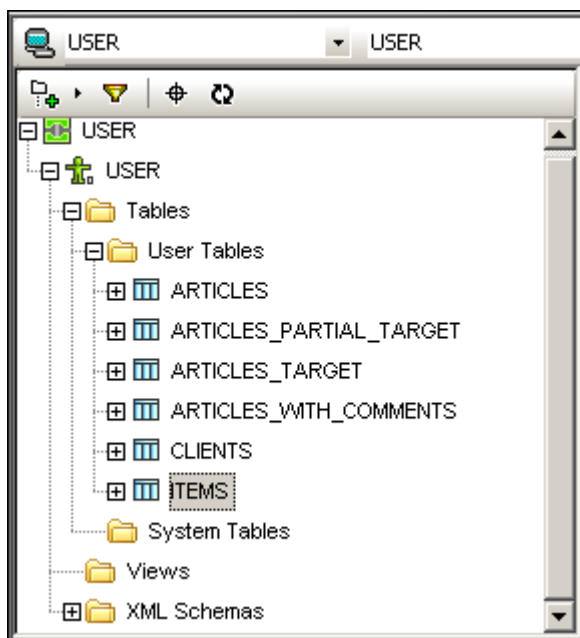
17.15.5 Datenbankabfrage - Browser-Fenster

Zu jeder der (vielen) verbundenen Datenquellen sehen Sie im **Browser**-Fenster eine Gesamtübersicht der Objekte in den einzelnen Datenbanken, einschließlich Datenbank Constraint-Informationen, z.B. Spalten als Primär- oder Sekundärschlüssel. In Datenbanken vom Typ IBM DB2 Version 9 werden im Browser zusätzlich registrierte XML-Schemas in einem separaten Ordner angezeigt.

Das Browser-Fenster kann angepasst werden um:

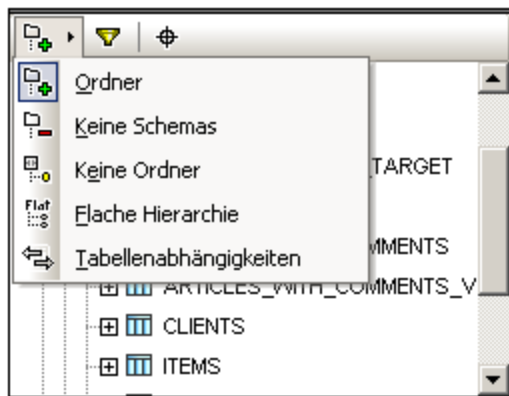
- bei Anzeige von Datenbankobjekten bestimmte Ordner-**Layouts** anzuzeigen
- bestimmte Objekte in der Datenbank mit Hilfe der Objektsuche zu **suchen**
- die Anzahl der angezeigten Objekte zu **filtern**
- das Root-Objekt der aktiven Datenquelle zu **aktualisieren**

Im Standardlayout "Ordner" werden Datenbankobjekte auf hierarchische Art angezeigt. Je nach ausgewähltem Objekt stehen unterschiedliche Kontextmenüoptionen zur Verfügung, wenn Sie mit der rechten Maustaste auf ein Datenelement klicken.



So wählen Sie ein Layout für den Browser aus:

- Klicken Sie im Browser auf die Schaltfläche "Layout"  und wählen Sie das Layout aus der Dropdown-Liste aus. Beachten Sie, dass sich die Schaltfläche ändert, je nachdem welches Layout ausgewählt ist.



Browser-Fenster- Layouts

Das Browser-Fenster enthält eine Reihe von vordefinierten Layouts zur Anzeige verschiedener Datenbankobjekte:

- Im Layout **Ordner** werden Datenbankobjekte auf Basis des Objekttyps in einer hierarchischen Struktur in Ordner unterteilt. Dies ist die Standardeinstellung.
- Das Layout **Keine Schemas** ähnelt dem Ordner-Layout, mit der Ausnahme, dass es keine Datenbankschema-Ordner gibt; Tabellen werden daher nicht nach Datenbankschemas gegliedert.
- Im Layout **Keine Ordner** werden Datenbankobjekte in einer Hierarchie ohne Verwendung von Ordnern angezeigt.
- Im Layout **Flach** werden Datenbankobjekte nach ihrem Typ auf der ersten hierarchischen Ebene geordnet. So werden Spalten z.B. anstatt in der entsprechenden Tabelle in einem separaten Ordner für Spalten angezeigt.
- Im Layout **Tabellenabhängigkeiten** werden Tabellen nach ihren Beziehungen zu anderen Tabellen gegliedert. Es gibt Kategorien für Tabellen mit Sekundärschlüsseln, durch Sekundärschlüssel referenzierte Tabellen und Tabellen, die in keiner Beziehung zu anderen Tabellen stehen.

So sortieren Sie Tabellen in Benutzer- und Systemtabellen:

1. Rechtsklicken Sie im Browser auf den Ordner "Tabellen". Daraufhin wird ein kontextsensitives Menü angezeigt.
2. Wählen Sie den Befehl **Nach Benutzer- und Systemtabellen sortieren**. Die Tabellen werden alphabetisch in die Ordner "Benutzertabellen" und "Systemtabellen" sortiert.

Bitte beachten Sie: Sie müssen sich in einem der Layouts Ordner, Keine Schemas oder Flach befinden, um diese Funktion aufrufen zu können.

So aktualisieren Sie das Root-Objekt der aktiven Datenquelle:

- Klicken Sie in der Symbolleiste auf die Schaltfläche .

Filtern und Suchen von Datenbankobjekten

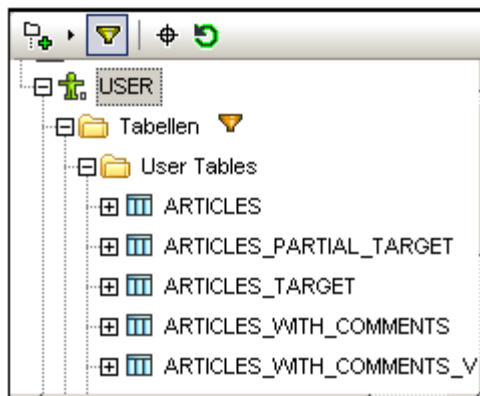
Im Browser können Sie Schemas, Tabellen und Ansichten nach dem Namen oder einem Teil des Namens filtern. Die Objekte werden während der Eingabe der Filterkriterien gefiltert. Standardmäßig wird die Groß- und Kleinschreibung bei der Filterung nicht berücksichtigt.

Bitte beachten Sie: Die Filter-Komponente funktioniert nicht, wenn Sie das Layout "Keine

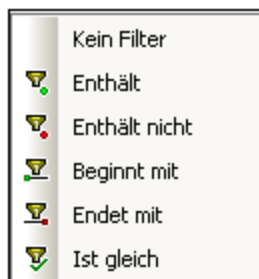
Filter" verwenden.

So filtern Sie Objekte im Browser:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche **Ordnerinhalt filtern** .
Neben allen Ordnern in derzeit ausgewählten Layout erscheinen Filtersymbole.

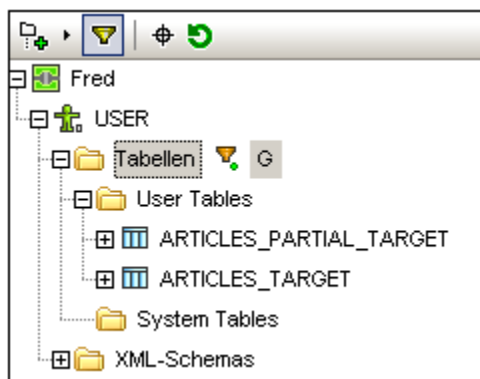


2. Klicken Sie auf das Filtersymbol neben dem gewünschten Ordner und wählen Sie im Popup-Menü die Filteroptionen fest, z.B. "Enthält".



Daraufhin wird neben dem Filtersymbol ein leeres Feld angezeigt.

3. Geben Sie hier den gesuchten String ein, z.B. "G". Das Ergebnis wird während der Eingabe angepasst.



Suchen von Datenbankobjekten

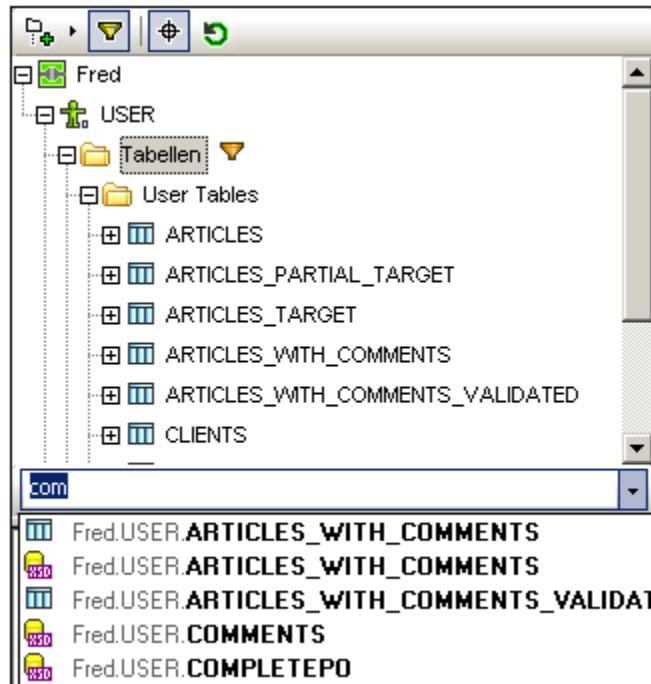
Um ein bestimmtes Datenbankobjekt nach seinem Namen zu suchen, verwenden Sie entweder Filterfunktionen oder die **Objektsuche** im Browser.

So suchen Sie Datenbankelemente mit Hilfe der Objektsuche:

1. Klicken Sie im Browser auf die Schaltfläche **Objektsuche** .

Daraufhin wird eine Dropdown-Liste am unteren Rand des Browsers angezeigt.

2. Geben Sie den gewünschten Suchstring ein, z.B. "com".
Wenn Sie auf den Dropdown-Pfeil klicken, werden alle Elemente, die den String enthalten, angezeigt.



3. Klicken Sie in der Liste auf das Objekt, um es im Browser anzuzeigen.

Kontextoptionen in der Browser-Ansicht

In der Browser-Ansicht stehen die folgenden Kontextmenüarten zur Verfügung:

- bei Rechtsklick auf das "Root"-Objekt
- bei Rechtsklick auf einen Ordner z.B. User tables
- bei Rechtsklick auf jede Art von Datenbankobjekt z.B: Tabellen ARTICLES

Wenn Sie mit der rechten Maustaste auf das Root-Objekt klicken, haben Sie die Möglichkeit, die Datenbank zu **aktualisieren**.

Wenn Sie mit der rechten Maustaste auf einen **Ordner** klicken, stehen immer dieselben Optionen zur Auswahl:

Erweitern | Gleichrangige | Children
Reduzieren | Gleichrangige | Children

Wenn Sie mit der rechten Maustaste auf ein **Datenbankobjekt** klicken, stehen folgende Optionen zur Verfügung:

In neuem SQL Editor anzeigen sowie die weiter unten beschriebenen Einträge des Untermenüs.

Bitte beachten Sie: Die Syntax der Anweisungen kann variieren je nachdem, welche Datenbank Sie verwenden. In den Beschreibungen unten wird Microsoft SQL Server 2000 als Beispiel verwendet. Verwenden Sie Umschalt + Klicken und Strg + Klick, um mehrere Datenbankobjekte auszuwählen.

Die folgenden Optionen stehen im Kontextmenü für **Tabellen** zur Verfügung:

- **Select:** Erstellt eine SELECT-Anweisung, mit der Daten aus allen Spalten der Quelltable abgerufen werden.
z.B. `SELECT "ID", "ARTICLE", "ARTICLE_AS_STRING" FROM "USER"."ARTICLES";`
- **Name:** Gibt den Namen der Tabelle zurück. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.
- **Pfad:** Gibt den vollständigen Pfad zu der/den Tabellen zurück, d.h., Datenquellename. Datenbankname.Schemaname.Tabellenname. Sie können auch mehrere Tabellen auswählen. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.

Die folgenden Optionen stehen im Kontextmenü für **Spalten** zur Verfügung:

- **Select:** Erstellt eine SELECT-Anweisung, mit der Daten aus der/den ausgewählten Spalte(n) der übergeordneten Tabelle abgerufen werden.
z.B. `SELECT "ARTICLE" FROM "USER"."ARTICLES";`
- **Name:** Gibt den Namen der ausgewählten Spalte zurück. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.
- **Pfad:** Gibt den vollständigen Pfad zur Spalte zurück, d.h., Datenquellename. Datenbankname.Schemaname.Tabellenname.Spaltenname. Sie können auch mehrere Spalten auswählen. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.

Die folgenden Optionen stehen im Kontextmenü für **Constraints** zur Verfügung:

- **Name:** Gibt den Namen des ausgewählten Constraint zurück. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.
- **Pfad:** Gibt den vollständigen Pfad zum Constraint zurück, d.h., Datenquellename. Datenbankname.Schemaname.Tabellenname.Constraint-Name. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.
z.B. `"USER"."ARTICLES_PARTIAL_TARGET"."CC1175533269606"`

Die folgenden Optionen stehen im Kontextmenü für **Indizes** zur Verfügung:

- **Name:** Gibt den Namen des ausgewählten Indexes zurück. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.
- **Pfad:** Gibt den vollständigen Pfad zum Index zurück, d.h., Datenquellename. Datenbankname.Schemaname.Tabellenname.Indexname. Die Namen werden auf durch Kommas getrennten separaten Zeilen ausgegeben.

17.15.6 Datenbankabfrage - Ergebnis- und Meldungsregister

Auf dem **Ergebnisregister** des SQL Editors sehen Sie den als Ergebnis der Datenbankabfrage abgerufenen Datensatz.

Amount: 200				
	ID	NAME	STATUS	CONTACTINFO
1	3227	Ella Kimpton-Smith	Gold	<?xml version="1.0" encoding="UTF-8" ?><...
2	8877	Chris Bontempo	Gold	<?xml version="1.0" encoding="UTF-8" ?><...
3	5681	Paula Lipenski	Standard	<?xml version="1.0" encoding="UTF-8" ?><...

Abruf fertig gestellt

Results Messages

Mapping **Datenbankabfrage** Ausgabe

Auswählen und Kopieren von Daten im Ergebnisfenster:

Es gibt verschiedene Methoden, um Daten in diesem Fenster auszuwählen. Sie können die Daten anschließend in andere Applikationen kopieren.

- Klicken Sie auf eine Spaltenüberschrift, um die Spaltendaten auszuwählen
- Klicken Sie auf eine Zeilennummer, um Zeilendaten zu markieren
- Klicken Sie auf einzelne Zellen

Wenn Sie beim Klicken die Strg-Taste gedrückt halten, können Sie mehrere Elemente auswählen. Wenn eine Spalte oder Zelle XML-Daten enthält, können diese Daten ebenfalls kopiert werden.

- Rechtsklicken Sie und wählen Sie im Kontextmenü den Befehl **Ausgewählte Zellen kopieren**.

Anmerkung: Sie können über das Kontextmenü Daten auch auswählen, **Auswahl | Zeile | Spalte | Alle**.

So sortieren Sie Daten in Ergebnisfenstern:

- Rechtsklicken Sie auf eine beliebige Stelle in der Spalte, die sortiert werden soll, und wählen Sie **Sortierung | Aufsteigend** oder **Absteigend**.
- Klicken Sie auf das Sortiersymbol in der Spalteüberschrift

	ID	NAME
1	3227	Ella Kim

Die Daten werden nach dem Inhalt der sortierten Spalte sortiert.

So stellen Sie die Standard-Sortierreihenfolge wieder her:

- Rechtsklicken Sie auf eine beliebige Stelle in der Tabelle und wählen Sie im Kontextmenü den Befehl **Sortierung | Standardeinstellung wiederherstellen**.

Symbolleistenoptionen - Abrufmodus

Das Ergebnisfenster enthält eine Symbolleiste, mittels der Sie zwischen den Ergebnissen und SQL-Anweisungen navigieren können und Teile von Datenbankdaten einfacher abrufen können.

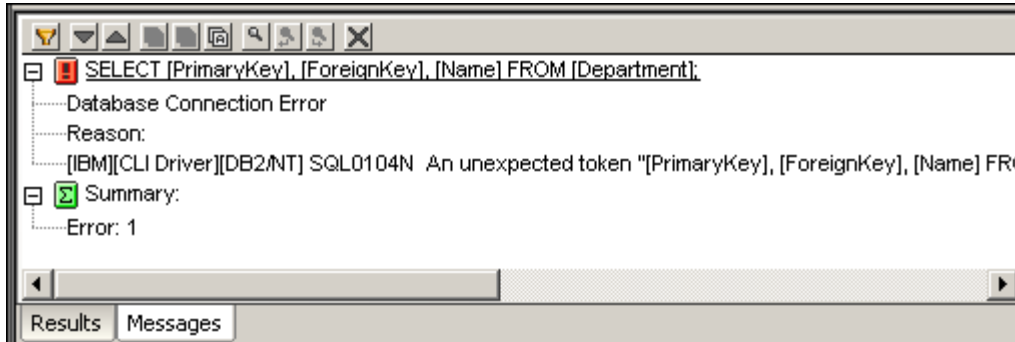


Suchen: Durchsucht den Input-String im Ergebnisfenster. Mit Hilfe der Funktionstaste F3 können Sie die Suche fortsetzen.



Gehe zu Anweisung: Geht zum SQL Editor-Fenster und markiert die Gruppe von SQL-Anweisungen, die das aktuelle Ergebnis erzeugt hat.

Auf dem **Meldungsregister** des SQL Editors werden bestimmte Informationen zur zuvor ausgeführten SQL-Anweisung sowie Fehler- und Warnmeldungen angezeigt.



Sie können zum Anpassen der Ansicht des Meldungsregisters verschiedene Filter verwenden oder das Fenster Zeile für Zeile mit Hilfe der Schaltflächen **Nächste** und **Vorherige** durchsuchen. Auf dem Meldungsregister steht außerdem ein Dialogfeld **Suchen** zur Verfügung sowie eine Reihe von Optionen zum Kopieren von Text in die Zwischenablage.

Symbolleistenoptionen

Das Meldungsregister enthält eine Symbolleiste, über die Sie innerhalb der Meldungen navigieren können und die Filter zum Ein- und Ausblenden bestimmter Teile der Meldung enthält.



Filter: Wenn Sie auf diese Schaltfläche klicken, wird ein Popup-Menü aufgerufen, in dem Sie einzelne Meldungsteile auswählen können (**Übersicht**, **Erfolg**, **Warnung**, **Fehler**). Des Weiteren können Sie alle oder keiner der Optionen mit einem einzigen Mausklick aktivieren bzw. deaktivieren. Wählen Sie einfach aus dem Popup-Menü eine der Optionen **Alle aktivieren** oder **Alle deaktivieren**.



Nächste: Springt zur nächsten Meldung und markiert sie.



Vorherige: Springt zur vorherigen Meldung und markiert sie.



Ausgewählte Meldung in die Zwischenablage kopieren



Ausgewählte Meldung einschließlich ihrer Subeinträge in die Zwischenablage kopieren



Alle Meldungen in die Zwischenablage kopieren



Suchen: Ruft das Dialogfeld **Suchen** auf.



Vorheriges suchen: Springt zur vorhergehenden Instanz des im Dialogfelds **Suchen** definierten String.



Weitersuchen: Springt zur nächsten Instanz des im Dialogfelds **Suchen** definierten

String.



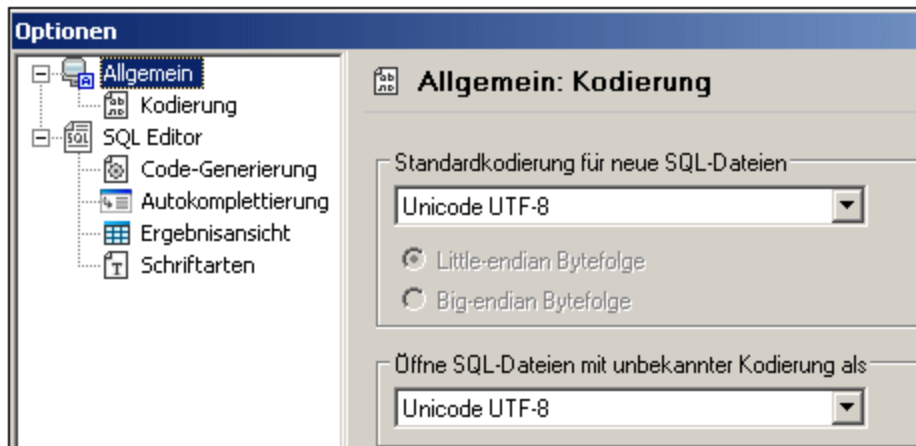
Löschen: Entfernt alle Meldungen vom Meldungsregister des SQL Editor-Fensters.

Bitte beachten Sie:

Dieselben Optionen stehen im Kontextmenü des Ergebnisfensters zur Verfügung.

17.15.7 Datenbankabfrage - Einstellungen

Im Abschnitt "Kodierung" des Dialogfelds **Optionen** können Sie verschiedene Kodierungsoptionen definieren.



Standardkodierung für neue SQL-Dateien

Definieren Sie die Standardkodierung für neue Dateien, sodass alle neuen Dokumente die Kodierungsspezifikation haben, die Sie hier definiert haben. Wenn als Standardkodierung eine 2- oder 4-Byte Kodierung (also UTF-16, UCS-2 oder UCS-4) als Standardkodierung ausgewählt wurde, können Sie auch zwischen Little Endian und Big Endian Bytefolge für die SQL-Dateien wählen.

Die Kodierung für bestehende Dateien wird natürlich immer beibehalten.

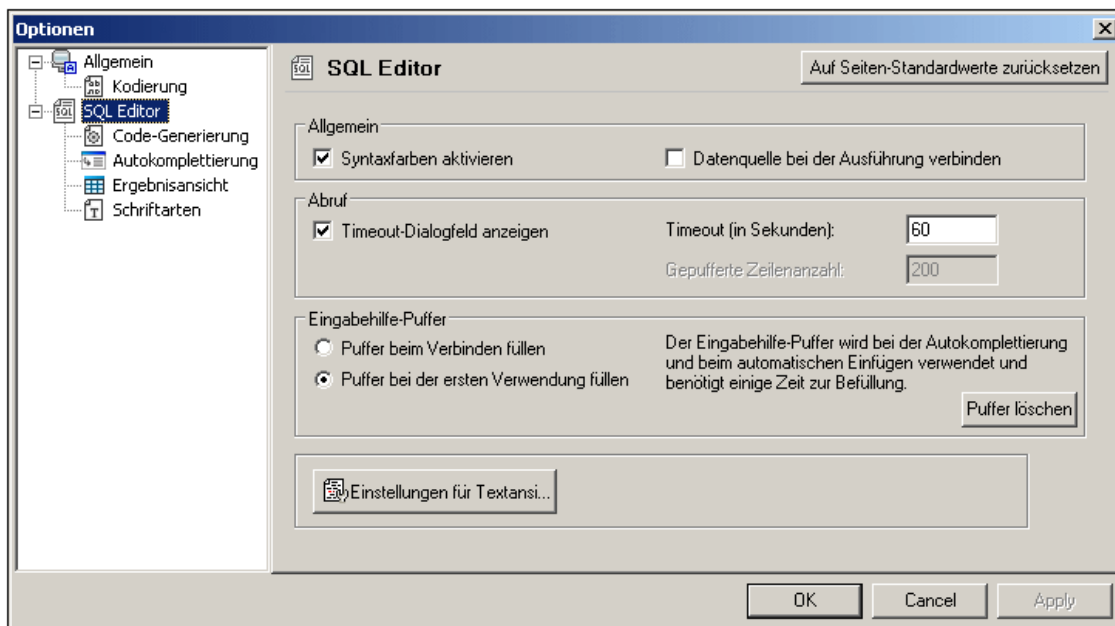
Öffne SQL-Dateien mit unbekannter Kodierung als

Sie können die Kodierung, mit der eine SQL-Datei geöffnet werden soll, auswählen, wenn keine Kodierungsspezifikation vorhanden ist oder wenn die Kodierung nicht ermittelt werden kann.

Bitte beachten Sie: SQL-Dateien ohne Kodierungsspezifikation werden mit einer UTF-8-Kodierung korrekt gespeichert.

SQL Editor Optionen

Auf der Hauptseite der SQL Editor-Optionen können Sie das optische Aussehen des Editors definieren und die Autokomplettierungsoptionen festlegen. Zusätzliche SQL Editor-Einstellungen werden unter den Optionen [Generierung](#), [Autokomplettierung](#), [Ergebnisansicht](#) und [Schriftarten](#) definiert.



Allgemein

Der SQL Editor bietet eine Autokomplettierungsfunktion, mit der Sie während der Eingabe aus einer Dropdown-Liste von SQL-Schlüsselwörtern und Datenbankobjektnamen auswählen können. Mit dem Kontrollkästchen "Autokomplettierung" werden die auf der Seite "Autokomplettierung" definierten Autokomplettierungseinstellungen aktiviert.

Syntaxfarben hebt die verschiedenen Elemente der SQL-Syntax mit Hilfe unterschiedlicher Farben hervor.

Passen Sie die **Tabulatorbreite** an, um die Anzahl der Leerzeichen zu definieren, die eingefügt werden sollen, wenn Sie im SQL Editor-Fenster die Tabulatortaste drücken.

Wenn Sie das Kontrollkästchen **Datenquelle bei der Ausführung verbinden** aktivieren, wird automatisch eine Verbindung zur entsprechenden Datenquelle hergestellt, wenn eine SQL-Datei ausgeführt wird und ihre Datenquelle nicht verbunden ist.

Abruf

Geben Sie die maximal für die SQL-Ausführung zulässige Zeit in Sekunden an.

Wenn Sie das Kontrollkästchen **Timeout-Dialogfeld anzeigen** aktivieren, können Sie die Timeout-Einstellungen ändern, wenn die zulässige Ausführungszeit überschritten wird.

Eingabehilfe-Puffer

Hier können Sie definieren, wie der Eingabehilfe-Puffer gefüllt werden soll - bei Herstellung der Verbindung oder erst wenn er das erste Mal benötigt wird.

Einstellungen für Textansicht

Hier können Sie die speziellen Einstellungen für die Textansicht definieren: Leisten, Register, visuelle Hilfen. Außerdem werden darin die Tastaturkürzel für die Navigation in der Textansicht angezeigt.

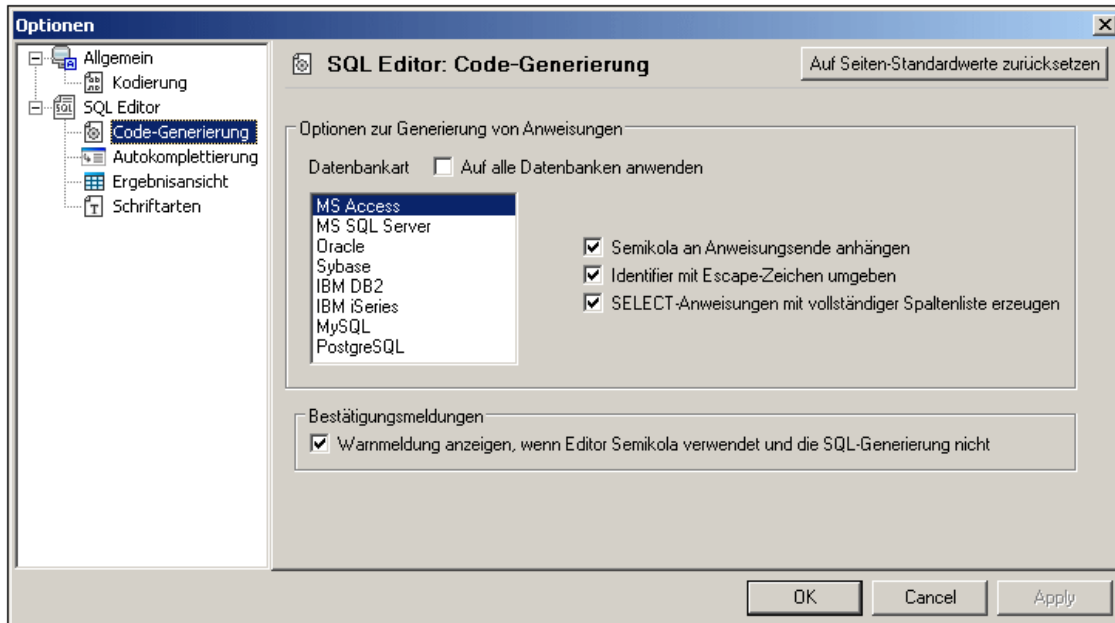
Generation

Im Abschnitt "Code-Generierung" des Dialogfelds **Optionen** können Sie die Syntax für die Generierung von Anweisungen für die verschiedenen Datenbanken definieren.

Wenn Sie in der Liste auf eine Datenbank klicken, können Sie mittels der drei Kontrollkästchen

auf der rechten Seite die Syntax definieren.

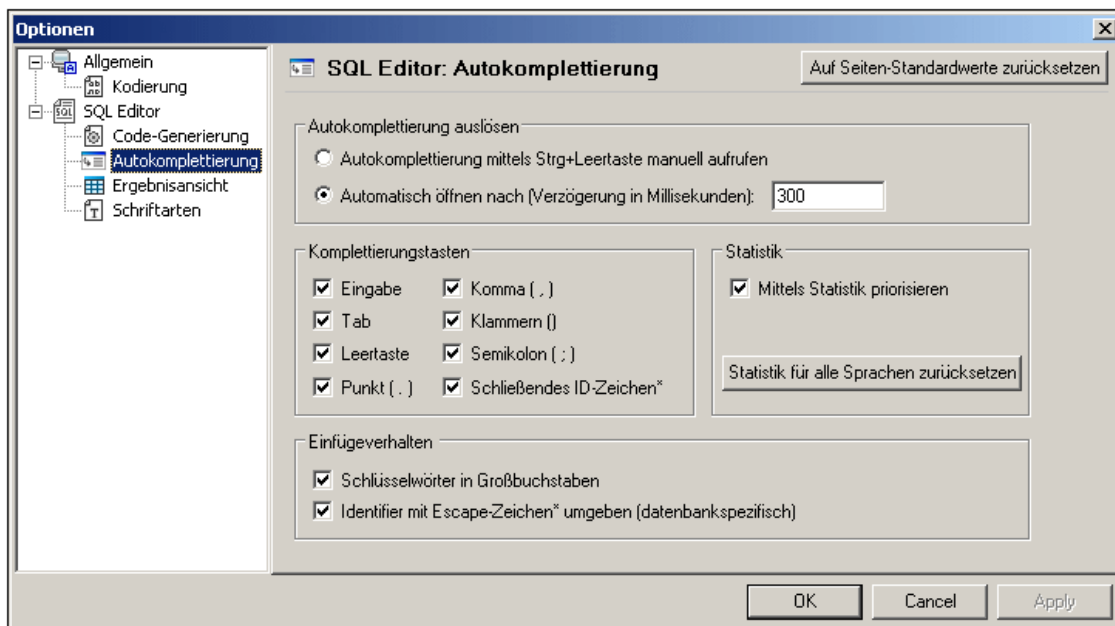
Wenn Sie das Kontrollkästchen "Auf alle Datenbanken anwenden" aktivieren, wird die Liste der Datenbanken ausgegraut und die Einstellungen in den Kontrollkästchen werden auf alle Datenbanken in der Liste angewendet.



Beachten Sie, dass die Bearbeitung von Daten in Oracle Datenbanken und IBM iSeries- und DB2-Datenbanken über eine JDBC-Verbindung nur möglich ist, wenn dieses Kontrollkästchen deaktiviert ist.

Autokomplettierung

Im Abschnitt "Autokomplettierung" des Dialogfelds **Optionen** können Sie die Autokomplettierungseinstellungen definieren.



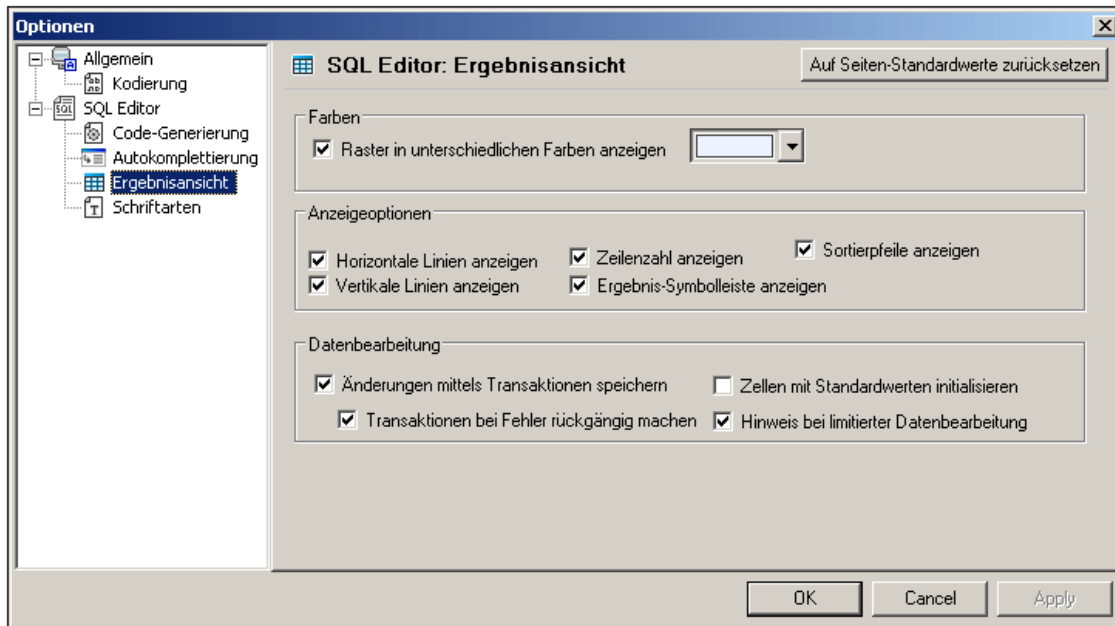
Die Autokomplettierung kann manuell oder automatisch aufgerufen werden.

Im Abschnitt "Komplettierungstasten" können Sie die Zeichen definieren, die bestimmte Schlüsselwörter einfügen und das Autokomplettierungsfenster schließen.

Unter "Einfügeverhalten" können Sie Groß-/Kleinschreibung definieren und ob die Identifier in Escape-Zeichen eingeschlossen werden sollen.

Ergebnisansicht

Im Abschnitt "Ergebnisansicht" des Dialogfelds **Optionen** können Sie einige Aspekte des Aussehens des Ergebnisfensters im SQL Editor konfigurieren.



Farben

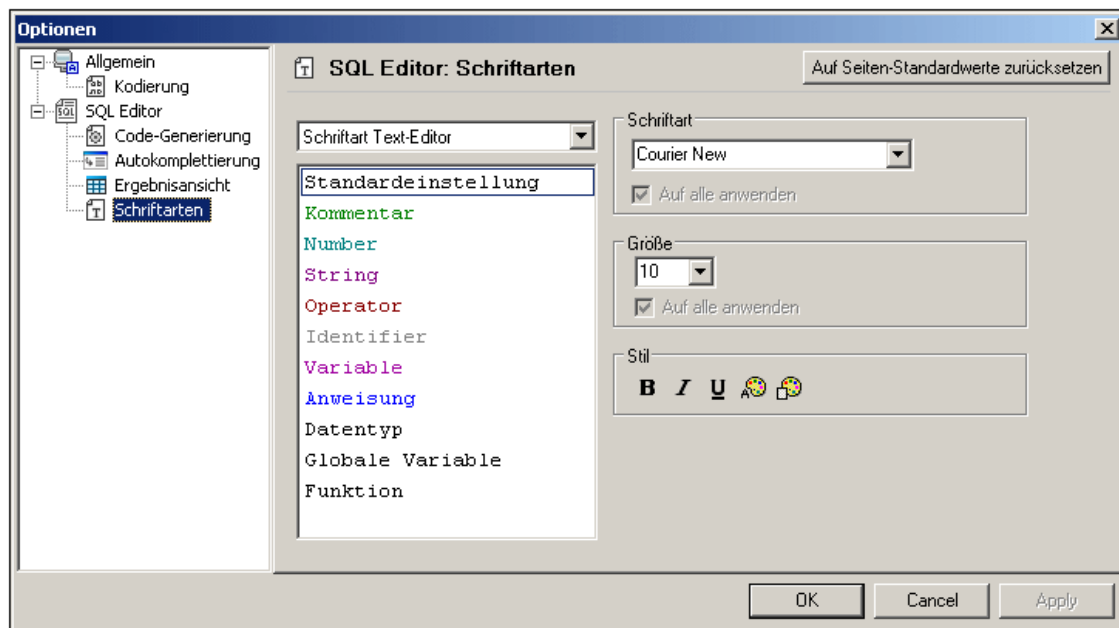
Sie können Zeilen auf dem Ergebnisregister als einfaches Raster oder abwechselnd als rote und weiße Zeilen anzeigen.

In der Gruppe **Anzeigeoptionen** können Sie definieren, wie horizontale und vertikale Linien sowie Zeilennummern und die **Ergebnis**-Symboleiste angezeigt werden. Sie können jede dieser Optionen durch Deaktivierung des jeweiligen Kontrollkästchens deaktivieren.

In der Gruppe **Datenbearbeitung** können Sie die Transaktionseinstellungen definieren, ob die Zellen mit Standardwerten befüllt werden sollen und ob ein Hinweis angezeigt werden soll, wenn die Datenbearbeitung eingeschränkt ist.

Code-Generierung

Im Abschnitt "Schriftarten" des Dialogfelds **Optionen** können Sie die Farbe und Schriftarteinstellungen einzelner Teile von SQL-Anweisungen definieren.



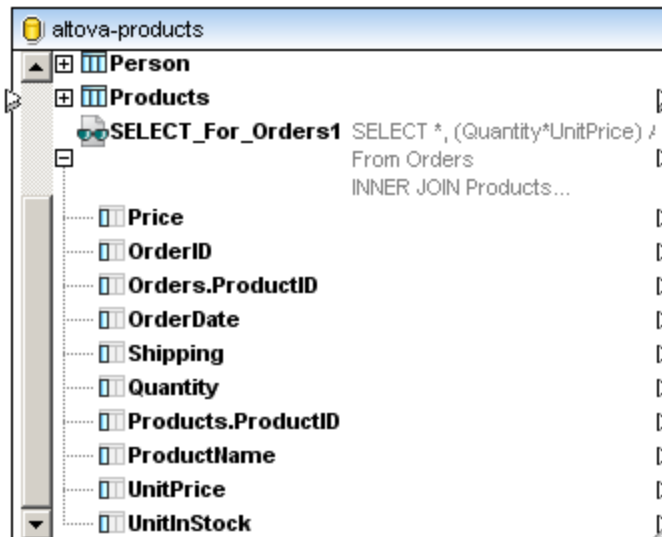
Die Schriftarteinstellungen, die im Listefeld "Schriftarteinstellungen" aufgelistet sind, sind Elemente von SQL-Anweisungen. Sie können die allgemeine Schriftart, den Stil und die Textgröße von Text, der im SQL-Editor angezeigt wird, auswählen. Beachten Sie, dass für alle Textarten dieselbe Schriftart und Größe verwendet wird.

Nur der Stil kann für einzelne Textarten geändert werden. Dadurch wird die Syntaxfärbungsfunktion aktiviert. Um die Originaleinstellungen wiederherzustellen, klicken Sie auf die Schaltfläche **Auf Standardeinstellungen zurücksetzen**.

17.16 SELECT-Anweisungen als virtuelle Tabellen

MapForce unterstützt nun auch die Erstellung von SQL SELECT-Anweisungen in Datenbankkomponenten. Es handelt sich hierbei um tabellenartige Strukturen, die die Felder der mit der SELECT-Anweisung generierten Tabellen enthalten. Diese Strukturen können anschließend wie jede andere in der Datenbank definierte Tabelle oder Ansicht als Datenquelle für das Mapping verwendet werden.

- Wenn Sie die inneren/äußeren **Joins** in der Select-Anweisung verwenden, werden Felder aller Tabellen in die Komponente inkludiert.
- Ausdrücke mit Entsprechungsnamen werden ebenfalls (unter Verwendung des SQL-Schlüsselworts "AS") als mapbare Datenelemente in der Komponente angezeigt,
- Auch Datenbankansichten können in der FROM-Klausel verwendet werden.
- SELECT-Anweisungskomponenten werden beim Einfügen von Datenbanktabellen in MapForce erstellt.



Anmerkung:

Die SELECT-Anweisung ist innerhalb der Komponente sichtbar. Um die Anzahl der Zeilen zu definieren, die von der Anweisung angezeigt werden sollen, wählen Sie die Menüoption **Extras | Optionen**, klicken Sie auf das Register "Allgemein" und geben Sie in der Gruppe "Mapping-Ansicht" die Anzahl der Zeilen ein.

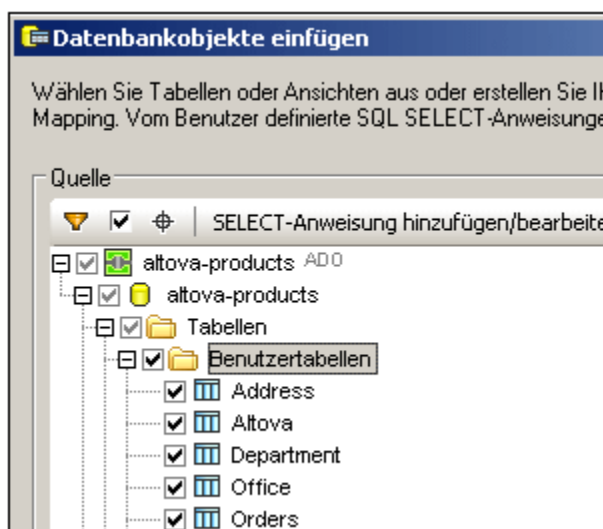
In diesem Abschnitt wird die Datei **altova-products.mdb** im Ordner **...\\MapForceExamples\\Tutorial** verwendet. Auch das Mapping **select-component.mfd**, das ein Beispiel enthält, befindet sich im selben Ordner.

17.16.1 Erstellen von SELECT-Anweisungen

In diesem Abschnitt wird die Datei **altova-products.mdb** aus dem Ordner ...
\\MapForceExamples\\Tutorial verwendet.

So erstellen Sie eine SELECT-Anweisung in einer Datenbankkomponente:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche **Datenbank einfügen** .
2. Wählen Sie das Optionsfeld **Microsoft Access** und klicken Sie anschließend auf "Weiter".
3. Klicken Sie auf die Schaltfläche "Durchsuchen" um die Datenbank auszuwählen, die Sie als Datenquelle verwenden möchten, in diesem Fall **altova-products.mdb** im Ordner ...**\\MapForceExamples\\Tutorial** und klicken Sie auf "Weiter".
4. Aktivieren Sie das Kontrollkästchen "Benutzertabellen" um alle Tabellen der Datenbank auszuwählen.

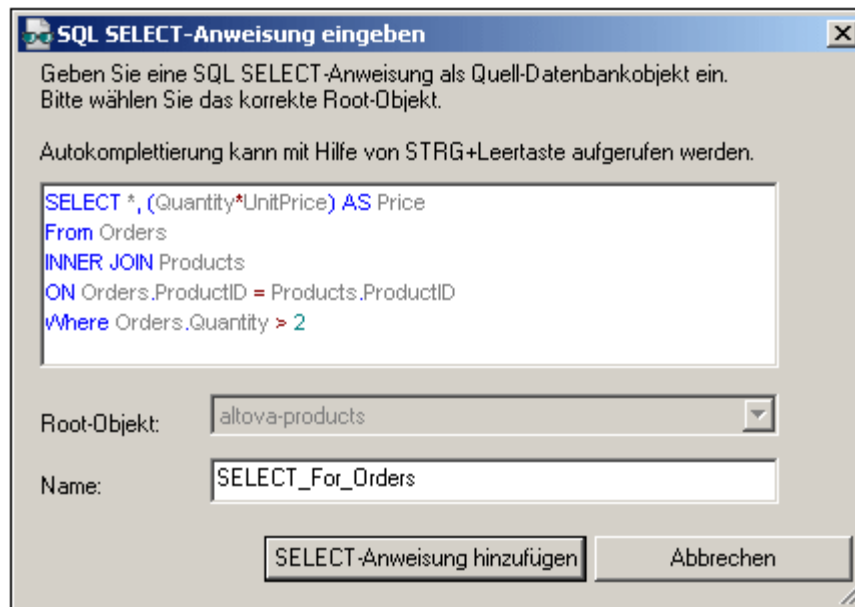


5. Klicken Sie mit der rechten Maustaste auf die Tabelle **Orders** und wählen Sie im Popup-Menü "SQL-Anweisung erzeugen und hinzufügen" aus (oder klicken Sie in der Symbolleiste auf die Schaltfläche "SELECT-Anweisung hinzufügen"). Im Dialogfeld steht bereits eine Standard-SELECT-Anweisung zur Verfügung.
6. Löschen Sie die Anweisung oder bearbeiten Sie sie Ihren Wünschen entsprechend. Im Beispiel wird die folgende Anweisung verwendet:

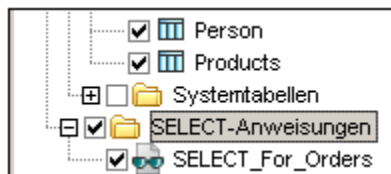
```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```

Bitte beachten Sie, dass alle berechneten Ausdrücke in der SELECT-Anweisung einen eindeutigen Entsprechungsnamen (wie in diesem Beispiel "AS Price") haben müssen, um als Datenelement für das Mapping zur Verfügung zu stehen.

Anmerkung: Wenn Sie über JDBC eine Verbindung zu einer Oracle- oder einer IBM DB2-Datenbank herstellen und zum Abrufen der Daten eine SELECT- Anweisung mit dem Befehl "Tabelle hinzufügen/entfernen" verwenden, so darf die SELECT-Anweisung keinen Doppelpunkt am Schluss enthalten.

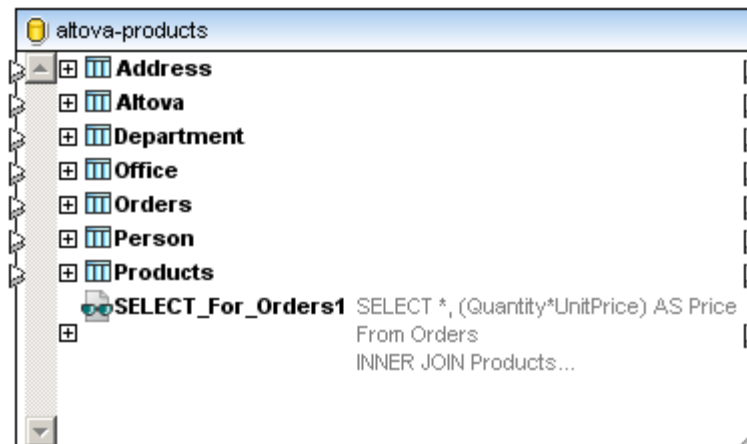


7. Klicken Sie auf die Schaltfläche **SELECT-Anweisung hinzufügen** um die Komponente einzufügen.



Die "virtuelle Tabelle" wurde nun im Dialogfeld zum Ordner SELECT-Anweisungen hinzugefügt.

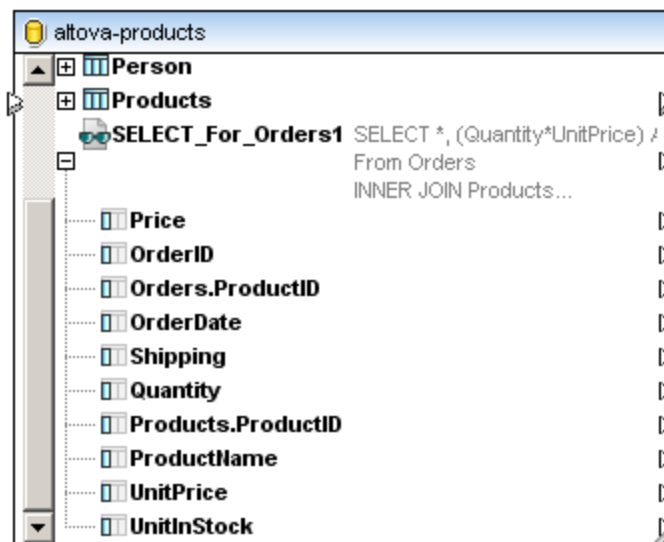
8. Klicken Sie auf OK, um die Komponente in den Mapping-Bereich einzufügen.



Was wurde eingefügt:

- Alle Tabellen der Altova-Datenbank von Address bis Products.
- Die virtuelle Tabelle **Select_For_orders** (Tabelle mit einer Gruppe von Ergebnissen) mit allen in der SELECT-Anweisung definierten Spalten oder Ausdrücken.

1. Klicken Sie auf das Erweiterungssymbol unterhalb des Eintrags **Select_For_Orders**.



Die Spaltenelemente der beiden durch die "inner join"-Anweisung definierten Tabellen stehen für das Mapping zur Verfügung.

Bitte beachten Sie:

Das Feld **Price** ist das Produkt der beiden Felder Quantity und UnitPrice und ist in Wirklichkeit ein **Alias**: `SELECT *, (Quantity*UnitPrice) AS Price`

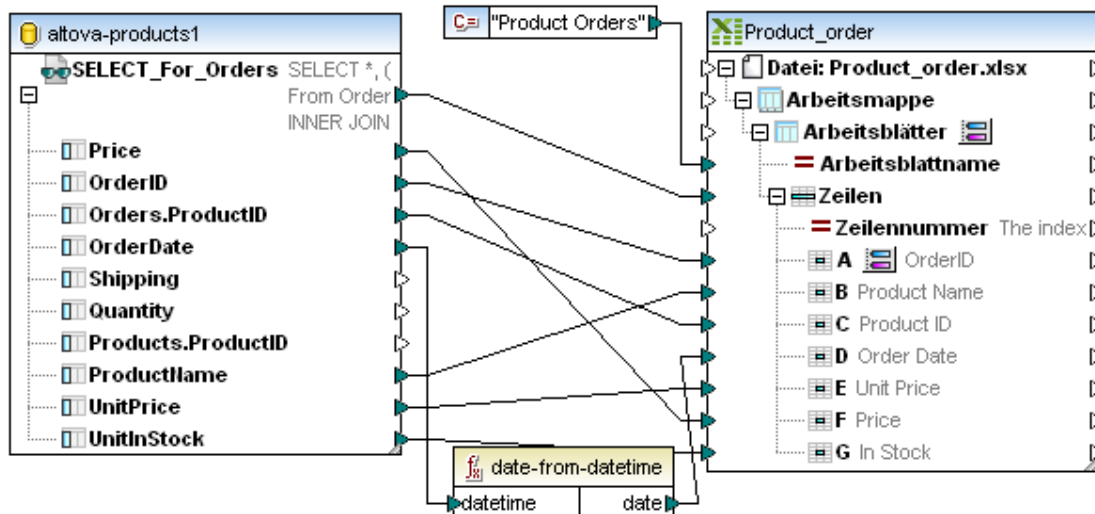
Die Select-Anweisung generiert alle Felder/Elemente, d.h. * und auch das Alias-Feld, welches durch das Komma getrennt ist.

17.16.2 Beispiel für eine SELECT-Anweisung

Im [vorigen Abschnitt](#) wurde beschrieben, wie eine SELECT-Anweisung in eine Datenbankkomponente eingefügt wird und welche Felder die Komponente enthält, wenn sie in den Mapping-Bereich eingefügt wird. In diesem Beispiel wird dieselbe Datenbank wie vorhin verwendet, nämlich **altova-products.mdb**.

Ziel dieses Mappings ist die Ausgabe bestimmter Bestelldaten aus der Datenbank und das Mappen dieser Daten auf eine OOXML Excel 2007 (und höher) Arbeitsmappe.

Öffnen Sie das Mapping **select-component.mfd** aus dem Ordner **...\\MapForceExamples\\Tutorial**.



Funktionsweise des Mappings:

- Es wird mittels einer inner join-Anweisung eine SELECT-Anweisungskomponente "altova-products" erstellt.
- Die Datenbankdaten werden auf eine leere Excel 2007 (und höher) Vorlagendatei gemappt. Nähere Informationen zum Mappen von/auf Excel 2007 (und höher) Arbeitsblätter finden Sie unter [Mappen von MS OOXML Excel 2007 \(und höher\)-Dateien](#)

Beachten Sie: Wenn Sie auf den Titel der Excel-Komponente doppelklicken, können Sie die Namen der Input oder Output-Datei der Excel-Komponente definieren.

Komponenteneinstellungen

Komponentenname: Product_order

Excel 2007+-Input-Datei

Excel 2007+-Output-Datei

☐ Fehler bei nicht übereinstimmendem Typ anzeigen

☒ Alle Dateipfade relativ zur MFD-Datei speichern

- Mit den **date-from-datetime**-Funktionen wird das Datenbankformat in das einfache Datenformat konvertiert.
- Wenn Sie auf die Schaltfläche "Ausgabe" klicken, wird eine Vorschau des Inhalts der Excel-Datei angezeigt.

	A	B	C	D	E	F	G	H
1	2	Layer designer	2	2008-02-06	5000	15000	no	
2	4	Visualizer	4	2007-12-03	3000	21000	yes	
3								
4								
5								
6								

Product Orders

Mapping Datenbankabfrage **Ausgabe**

Kapitel 18

Mappen von CSV und Textdateien

18 Mappen von CSV und Textdateien

MapForce unterstützt das Mappen von Flat File-Formaten, also von CSV- und Textdateien sowohl als Quell- als auch Zielkomponente. Bitte beachten Sie, dass Sie für die Mapping-Ausgabe eine der Programmiersprachen auswählen müssen (Java, C# oder C++) oder BUILTIN, um mit Text-Dateien arbeiten zu können.

Unterstützte Textdateiformate sind:

- CSV-Dateien (kommagetrennte Werte) - auch für andere Trennzeichen als Kommas
- FLF-Dateien (Fixed-Length Fields)
- andere Textdateien, deren Struktur in einer FlexText Vorlage definiert ist (nur Enterprise Edition)

Dieses bidirektionale Mapping unterstützt:

- XML-Schema auf Flat File-Formate und umgekehrt
- Datenbank auf Flat File-Formate und umgekehrt
- UN/EDIFACT und ANSI X12 von/auf Flat File-Formate oder Datenbanken
- eine FlexText-Vorlage, in der die im FlexText -Modul definierte Dateistruktur und der Inhalt definiert ist und als Komponente in ein Mapping eingefügt wird. Nähere Informationen siehe [MapForce FlexText](#)

Es gibt zwei Methoden, um gemappte Flat File-Daten zu generieren/speichern:

- Durch Klicken auf die Schaltfläche "Ausgabe", woraufhin mit Hilfe des Built-In-Ausführungsprozessors eine Vorschau generiert wird. Wählen Sie die Menüoption **Ausgabe | Ausgabedatei speichern** oder klicken Sie auf das Symbol , um das Ergebnis zu speichern.
- Durch Auswahl von **Datei | Code generieren in | Java, C# oder C++** und anschließendes Kompilieren und Ausführen des generierten Codes.

Bitte beachten Sie:

Alle der folgenden Beispiele, in denen CSV-Dateien als Quell- oder Zielkomponenten verwendet werden, können auch mit Textdateien mit fester Länge durchgeführt werden. Der einzige Unterschied ist, dass die Feldlängen manuell definiert werden müssen. Informationen zum Definieren von Feldlängen finden Sie unter "[Mappen von Textdateien mit fester Länge](#)".

18.1 Mappen von CSV-Dateien auf XML

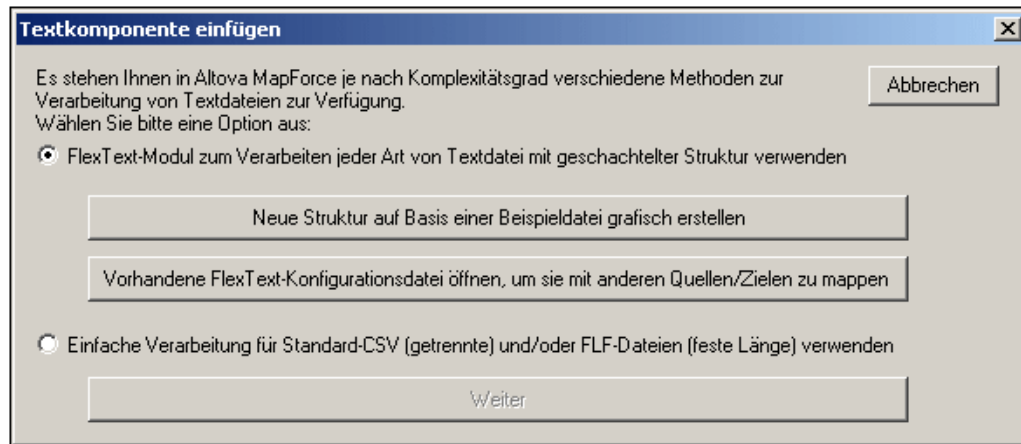
In diesem Beispiel wird eine einfache CSV-Datei auf eine XML-Datei gemappt, die auf der Schema-Datei MFCompany.xsd basiert. Alle im folgenden Beispiel verwendeten Dateien liegen im Ordner ...\\MapForceExamples\\Tutorial\\.

- Es muss eine der Programmiersprachen **Java**, **C#**, **C++** oder **BUILTIN** durch Aktivierung des entsprechenden Symbols in der Symbolleiste ausgewählt sein.
1. Wählen Sie die Menüoption **Einfügen | Textdatei** oder klicken Sie auf das Symbol



"Textdatei einfügen"

Daraufhin wird das Dialogfeld "Textkomponente einfügen" geöffnet.



Aktivieren Sie das Optionsfeld **Einfache Verarbeitung für ...** und klicken Sie auf die Schaltfläche **Weiter**.

Daraufhin wird das Dialogfeld "Text importieren / exportieren" geöffnet, wo Sie den Dateityp auswählen können, mit dem Sie arbeiten möchten: CSV oder Dateien mit "fester" Länge. Standardmäßig ist das Optionsfeld CSV aktiv.

Komponenteneinstellungen

Komponentenname:

OK
Abbrechen

Eingabe- / Ausgabedatei

Eingabedatei

Ausgabedatei

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- /Ausgabekodierung

Kodierungsname:

Bytefolge: ☐ Bytefolge-Markierung inkl.

CSV Einstellungen

Feldbegrenzung: ☐ Tab ☐ Semikolon ☒ Komma ☐ Leerz. ☐ Benutzerdef.

Textbegrenzung: ☐ Keine ☐ ' ☒ " ☐ CSV ☐ Feste Länge

☐ Erste Zeile enthält Feldnamen

☒ Leere Felder als nicht vorhanden behandeln

Field1	Field2	Field3	Field4	Field5
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Frank	Further	471	f.further@nanonull.com	Accounts Receivabl
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager

Feld anhängen Feld einfügen Feld löschen << >>

- Klicken Sie auf die Schaltfläche **Eingabedatei** und wählen Sie die CSV-Datei aus, z.B. Altova_csv.csv. Der Inhalt der Datei wird nun im Vorschauenfenster angezeigt. Beachten Sie, dass im Vorschauenfenster nur die ersten 20 Zeilen der Textdatei angezeigt werden.
- Klicken Sie in die Spaltenüberschrift Field1 und ändern Sie den Text z.B. in First-name. Ändern Sie auf die gleiche Weise die anderen Überschriften in Last-name, Tel.-extension, Email und Position.

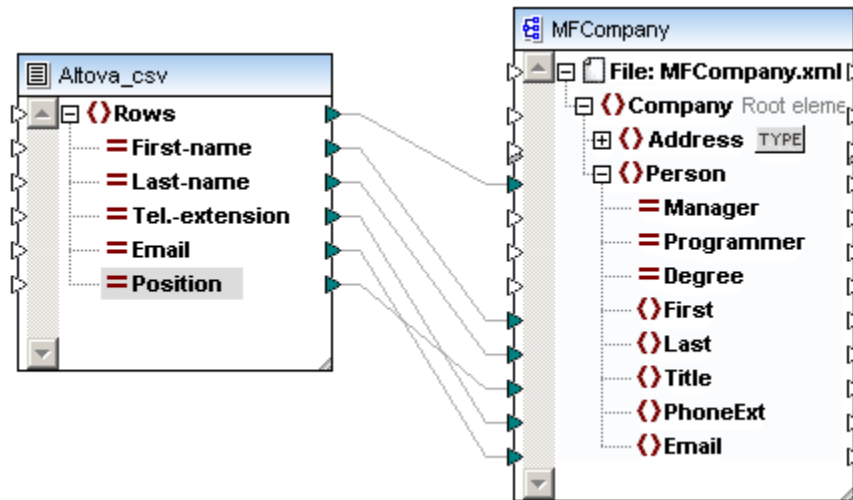
First-name	Last-name	Tel.-extension	Email	Position.
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Mana
Frank	Further	471	f.further@nanonull.com	Accounts Re
Loby	Matise	963	l.matise@nanonull.com	Accounting M
Joe	Firstbread	621	j.firstbread@nanonull.com	Marketing Me
Susi	Sanna	753	s.sanna@nanonull.com	Art Director

Bitte beachten Sie:

Durch Drücken der **Tabulatortaste** können Sie der Reihe nach von einem Feld zum nächsten springen: header1, field type1, header2 usw.

- Wenn Sie mit den Einstellungen zufrieden sind, klicken Sie auf OK.

5. Klicken Sie (in der angezeigten Eingabeaufforderung) auf die Schaltfläche "Komponentennamen ändern", um den Komponentennamen zu definieren. Die CSV-Komponente wird nun im Mapping angezeigt.
6. Wählen Sie die Menüoption **Einfügen | XML-Schema / Datei** und anschließend die Datei **MFCompany.xsd**.
7. Wenn Sie gefragt werden, ob Sie eine XML-Beispieldatei angeben möchten, klicken Sie auf "Überspringen" und wählen Sie Company als das Root-Element aus.



8. Mappen Sie die entsprechenden Datenelemente der beiden Komponenten so, dass das Datenelement **Rows** im Zielschema auf das Datenelement **Person** gemappt wird und klicken Sie anschließend auf die Schaltfläche Ausgabe, um das Ergebnis zu sehen.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3  <Person Manager="true">
4  <First>Vernon</First>
5  <Last>Callaby</Last>
6  <PhoneExt>582</PhoneExt>
7  <Email>v.callaby@nanonull.com</Email>
8  </Person>
9  <Person Manager="true">
10 <First>Frank</First>
11 <Last>Further</Last>
12 <PhoneExt>471</PhoneExt>
13 <Email>f.further@nanonull.com</Email>
14 </Person>
15 <Person Manager="true">

```

Die Daten aus der CSV-Datei wurden erfolgreich auf eine XML-Datei gemappt.

Bitte beachten Sie:

Der Konnektor zwischen dem Datenelement **Rows** in der CSV-Komponente und dem Datenelement **Person** im Schema ist unbedingt notwendig, da damit definiert wird, durch welches Element iteriert werden soll. Für jedes Row-Element in der CSV-Datei wird ein neues Person-Element in der XML-Ausgabedatei erstellt.

Beispiele dafür, wie sich das Datenelement **Rows** auf die Ausgabe auswirkt, wenn Sie **auf** eine CSV-Datei oder eine Datei mit fester Länge mappen, finden Sie auf den folgenden Seiten.

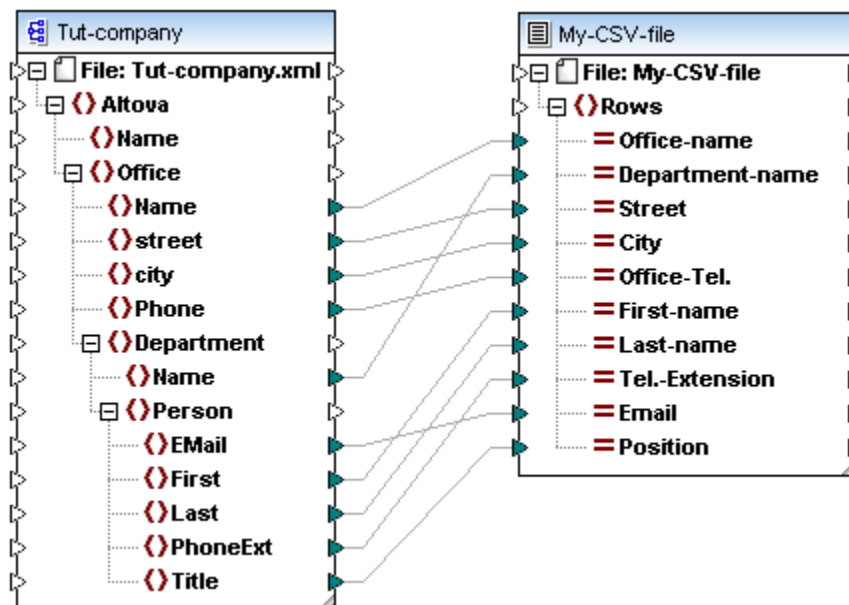
18.2 Mappen von XML auf CSV, Iterieren durch Datenelemente

Sie finden dieses Beispiel im Ordner ...\\MapForceExamples\\Tutorial\\ unter **Tut-xml2csv.mfd**.

- **Tut-company.xsd** und **Tut-company.xml** sind das Quellschema bzw. die XML-Datenquelle.
- "My-CSV-file" ist die Textdateikomponente. Der Name wird im Feld "Eingabedatei" des Dialogfelds "Text importieren / exportieren" eingegeben.

Das Mapping-Beispiel dient nur zu Demonstrationszwecken und ist kein reales Szenario.

Im unten gezeigten Diagramm sehen Sie, wie man normalerweise vorgehen würde, um eine XML-Datei auf eine CSV-Datei zu mappen.

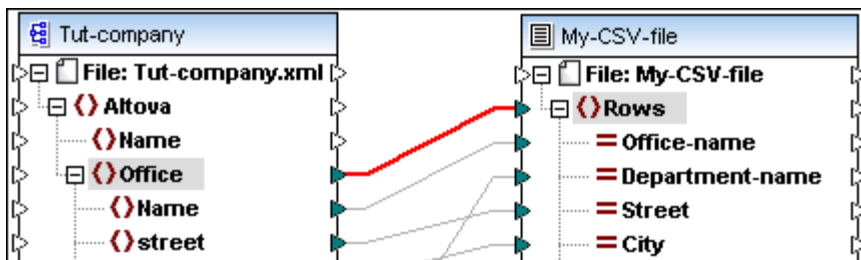


Wenn Sie auf die Schaltfläche Ausgabe klicken, sehen Sie das unten gezeigte Ergebnis, was wohl nicht wirklich Ihren Erwartungen entspricht, da wir nur die Ausgabe für ein Büro sehen.

```
1 "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
2
```

Um durch alle Büros zu iterieren und die Ausgabe in der CSV-Datei zu sehen, müssen Sie Office mit Rows verbinden. Dies bedeutet: Für jedes Datenelement "Office" in der XML-Quelldatei muss eine Zeile "Row" in der CSV-Zielfeile erstellt werden. Sie können in MapForce das Feld oder Datenelement angeben, das als "Root"/Iterator für die Ausgabe verwendet wird, also in diesem Fall das Datenelement **"Rows"**.

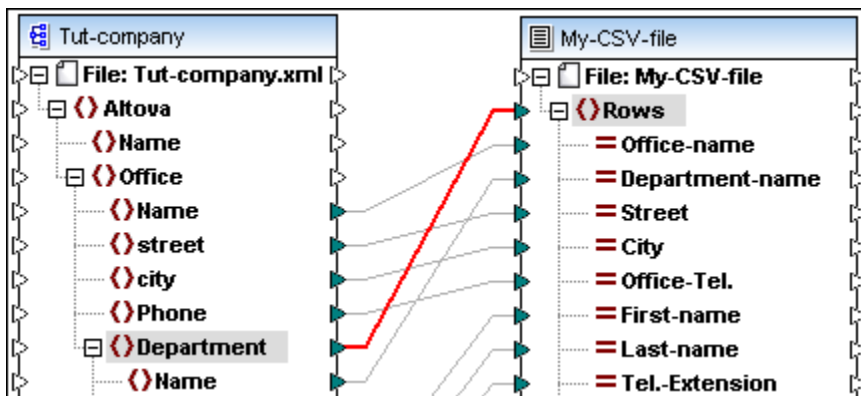
Wenn Sie das Datenelement **Office** auf **Rows** mappen, werden alle einzelnen Büros (und gemappten Datenelemente) ausgegeben.



Die Datenelemente "Office" werden in der Reihenfolge ausgegeben, in der sie in der Quelldatei aufscheinen.

```
1 "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,558833
2 "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Otto
3
```

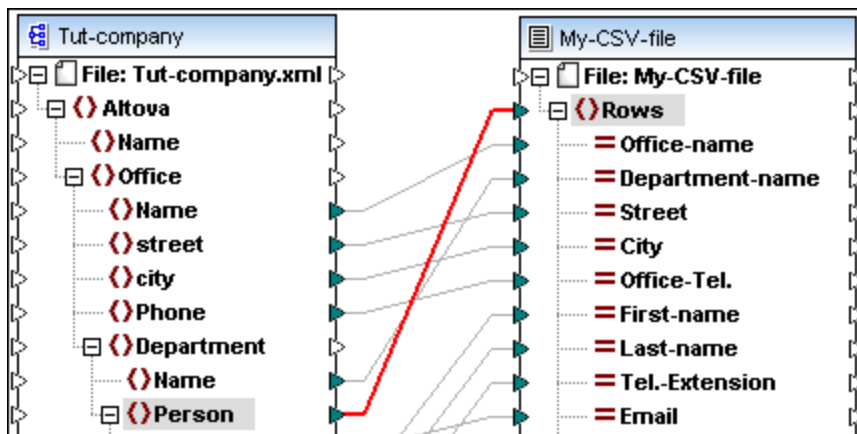
Wenn Sie **Department** auf das Datenelement **Rows** mappen, werden alle Abteilungen (Departments) ausgegeben.



Die Abteilungen werden für jedes Büro (Office) in der Reihenfolge der Quelldatei ausgegeben.

```
1 "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clo
2 "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,558
3 "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,K
4 "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,5588339
5 "Microtech Partners, Inc.",Admin,Perro Bvd 1324,Ottowa,3549202,
6 "Microtech Partners, Inc.",Sales and Marketing,Perro Bvd 1324,0
7 "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
8
```

Wenn Sie **Person** auf das Datenelement **Rows** mappen, werden alle Personen ausgegeben.



Die Personen werden in der Reihenfolge der Quelldatei ausgegeben, d.h. jede Person in jeder Abteilung für jedes Büro.

1	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Albert,Aldrich,582,A.Ald
2	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Bert,Bander,471,b.bander
3	"Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clovis,963,c.clovi
4	"Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Dave,Durne
5	"Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Eve,Ellas,
6	"Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Fred,Fortunas,95
7	"Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Gerry,Gundall,65

18.3 Hierarchien für CSV-Dateien und Dateien mit fester Länge

Sie finden dieses Beispiel im Ordner [...\\MapForceExamples\\Tutorial\\](#) unter **Tut-headerDetail.mfd**.

In diesem Beispiel wurde eine CSV-Datei mit Feldern verwendet, die die spezifischen Datensatztypen definieren. Die Datei hat das folgende Format:

- Field 1: H definiert einen Header-Datensatz und D einen Detail-Datensatz.
- Field 2: Ein gemeinsamer Schlüssel sowohl für Header- als auch Detail-Datensätze.
- Jeder Header/Detail-Datensatz befindet sich in einer eigenen Zeile.

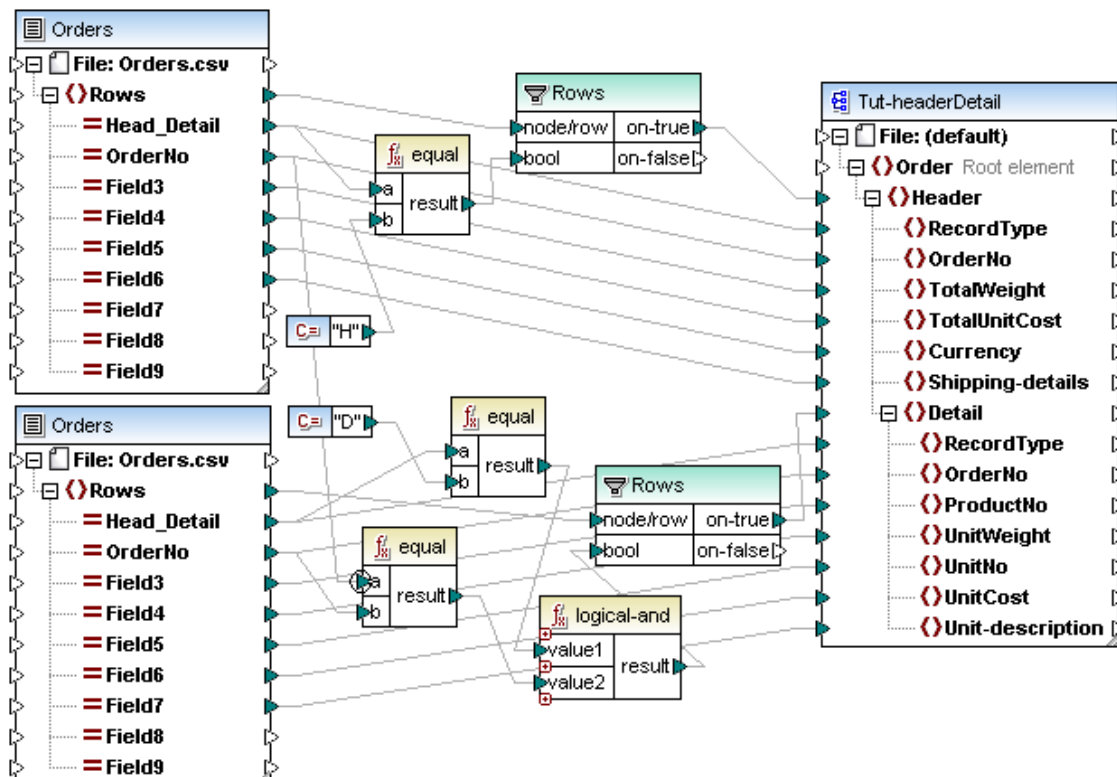
Erstellen von hierarchischen XML-Strukturen anhand von Flat Files mit Hilfe von "Schlüssel"-Feldern

Nachfolgend sehen Sie den Inhalt der Datei Orders.csv.

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

Ziel dieses Mapping ist:

- die Flat File CSV-Datei auf eine hierarchische XML-Datei zu mappen und
- die mit einem H gekennzeichneten Header-Datensätze zu filtern und
- die entsprechenden Detail-Datensätze, die mit einem D gekennzeichnet sind, mit den einzelnen Header-Datensätzen zu verknüpfen.



Damit dies möglich ist, müssen die Header- und Detail-Datensätze ein gemeinsames Feld haben. In diesem Fall handelt es sich beim gemeinsamen Feld/Schlüssel um das zweite Feld in

der CSV-Datei, nämlich **OrderNo**. In der CSV-Datei enthalten sowohl der erste Header-Datensatz als auch die folgenden beiden Detail-Datensätze den gemeinsamen Wert 111.

Anmerkungen zum Mapping:

Die Datei Orders.csv wurde zweimal eingefügt, um das Mapping intuitiver zu machen.

Die Schema-Datei **Tut-headerDetail.xsd** hat eine hierarchische Struktur: "Order" ist das Root-Element, das das Child-Element "Header" enthält, das wiederum das Child-Element "Detail" enthält.

Die erste Orders.csv-Datei liefert die **Header**-Datensätze (und alle gemappten Felder) für das Header-Datenelement in der Schema-Zieldatei. Mit Hilfe der filter-Komponente werden alle Datensätze mit Ausnahme derer, die mit H beginnen herausgefiltert. Das **Rows**-Datenelement liefert die gefilterten Datensätze für das Header-Datenelement in der Schema-Datei.

Die zweite Orders.csv-Datei liefert die **Detail**-Datensätze (und alle gemappten Felder), indem die Detail-Datensätze, die mit dem OrderNo-Schlüssel des Header-Datensatzes übereinstimmen, herausgefiltert werden. Dies geschieht folgendermaßen:

- Das **OrderNo**-Feld des Header-Datensatzes wird mit Hilfe der **equal**-Funktion (der [Prioritätskontext](#) wird aus Gründen der Effizienz für den Parameter **a** festgelegt) mit demselben Feld der Details-Datensätze verglichen.
- Die **Logical-and**-Funktion wird verwendet, um nur diejenigen Detail-Datensätze weiterzugeben, die dasselbe OrderNo-Feld enthalten, wie der Header-Datensatz.

Über den on-true-Parameter der filter-Komponente liefert das Datenelement **Rows** diese gefilterten Datensätze an die Datenelemente "Header" und "Detail" in der Schema-Datei.

Wenn Sie auf die Schaltfläche "Ausgabe" klicken, wird die unten gezeigte XML-Datei erzeugt. Jeder Header-Datensatz enthält die dazugehörigen Daten sowie alle damit verknüpften Detail-Datensätze mit derselben Bestellnummer (OrderNo).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="
3  <Header>
4      <RecordType>H</RecordType>
5      <OrderNo>111</OrderNo>
6      <TotalWeight>332.1</TotalWeight>
7      <TotalUnitCost>22537.7</TotalUnitCost>
8      <Currency/>
9      <Shipping-details>Container ship</Shipping-details>
10     <Detail>
11         <RecordType>D</RecordType>
12         <OrderNo>111</OrderNo>
13         <ProductNo>A-1579-227</ProductNo>
14         <UnitWeight>10</UnitWeight>
15         <UnitNo>3</UnitNo>
16         <UnitCost>400</UnitCost>
17         <Unit-description>Microtome</Unit-description>
18     </Detail>
19     <Detail>
20         <RecordType>D</RecordType>
21         <OrderNo>111</OrderNo>
22         <ProductNo>B-152-427</ProductNo>
23         <UnitWeight>7</UnitWeight>
24         <UnitNo>6</UnitNo>
25         <UnitCost>1200</UnitCost>
26         <Unit-description>Miscellaneous</Unit-description>
27     </Detail>
28 </Header>
```

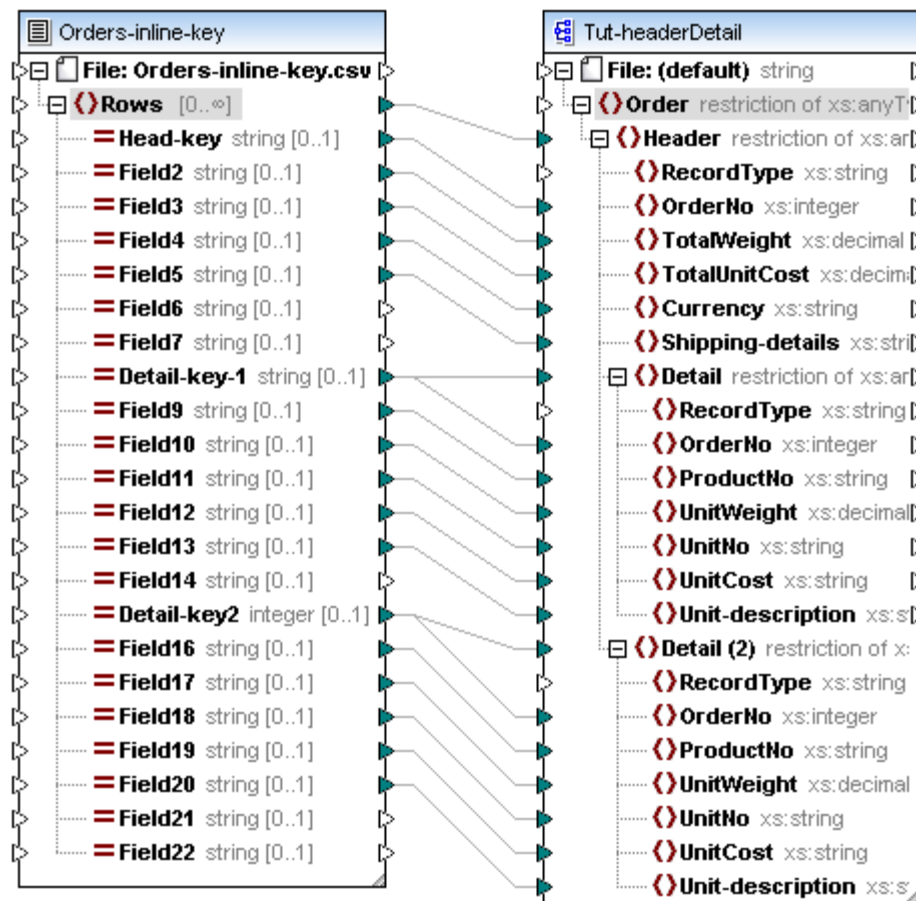
Im zweiten Beispiel wird eine etwas veränderte CSV-Datei verwendet. Sie finden das Beispiel im Ordner **...MapForceExamples\Tutorial** unter dem Namen **Head-detail-inline.mfd**. Im Unterschied zum ersten Beispiel:

- ist hier keine Datensatzkennung (H, or D) vorhanden
- Es gibt jedoch weiterhin ein gemeinsames Schlüsselfeld für den Header- und die Detail-Datensätze (Head-key, Detail-key...), nämlich das erste Feld in der CSV-Datei. Das Feld wird auf OrderNo im Zielschema gemappt.
- "Header" und alle entsprechenden "Detail"-Felder befinden sich alle in derselben Zeile.

```
111,332.1,22537.7,,Container ship,,,111,A-1579-227,10,3,400,Microtome,,111,B-15
222,978.4,7563,1,,Air freight,,,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

Bitte beachten Sie:

- Die Schlüsselfelder werden auf die entsprechenden OrderNo-Datenelemente im Zielschema gemappt.
- Das Datenelement "Detail" in der Schema-Zielfeile wurde dupliziert und wird als **Detail (2)** angezeigt, damit Sie den zweiten Satz von Detail-Datensätzen auf das richtige Datenelement mappen können.
- Als Ergebnis dieses Mappings erhalten Sie in etwa dieselbe XML-Datei, wie im obigen Beispiel.



18.4 CSV-Dateioptionen

Rechtsklicken Sie auf die Komponente **Orders_csv** und wählen Sie **Einstellungen**, um das Dialogfeld zu öffnen.

CSV Textimport/-Export-Einstellungen:

Wenn Sie in diesem Dialogfeld Feldformate definieren, wird der Typ der jeweiligen Felder automatisch überprüft. Wenn die Input-Daten und das hier definierte Feldformat nicht übereinstimmen, werden die Daten rot markiert. Wenn Sie z.B. Field2 von "string" in "integer" ändern, erscheinen alle Nachnamen in dieser Spalte rot markiert.

Bitte beachten Sie:

Welche Feldtypen man für eine bestimmte Spalte auswählen kann, ist abhängig von den Standard-XML-Schema-Datentypen. So ist z.B. der Datentyp in der Form: JJJJ-MM-TT.

Komponenteneinstellungen

Komponentenname: Orders

Eingabe- / Ausgabedatei: \\My Documents\\Altova\\MapForce2012\\MapForceExamples\\Tutorial\\Orders.csv

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- / Ausgabekodierung: Kodierungsname: Unicode UTF-8, Bytefolge: Little Endian, ☐ Bytefolge-Markierung inkl.

CSV Einstellungen: Feldbegrenzung: Komma, Textbegrenzung: CSV, ☐ Erste Zeile enthält Feldnamen, ☒ Leere Felder als nicht vorhanden behandeln

Field1	Field2	Field3	Field4	Field5	Field6	Field7	Field8
string	string	string	string	string	string	string	string
H	111	332.1	22537.7		Container ship		
D	111	A-1579-227	10	3	400	Microtome	
D	111	B-152-427	7	6	1200	Miscellaneous	

Feld anhängen, Feld einfügen, Feld löschen, <<, >>

Eingabedatei:

Wählen Sie die CSV-Datei, die als Quelldatei für diese Komponente verwendet werden soll. Dieser Name wird zum Lesen der Beispieldaten und Feldinformationen, für die Ausgabevorschau und für die Codegenerierung verwendet.

Bitte beachten Sie:

Dieses Feld kann leer bleiben, wenn Sie die Textdateikomponente als **Ziel** für Ihr Mapping verwenden. In diesem Fall wird als Standard-Encoding automatisch UTF-8 ausgewählt. Sie können den Feldtyp, die Feldnamen, die Formatierung usw. definieren. Klicken Sie anschließend auf OK, um eine Ziel-Textdatei zu erstellen.

Wenn Sie auf die Schaltfläche **Ausgabe** klicken, können Sie diese Textdatei mitsamt ihrem gemappten Inhalt durch Klicken auf das Symbol "Generierte Ausgabe

speichern..."  **speichern**.

Wenn Sie in dieses Textfeld einen Namen eingeben (ohne Dateierweiterung), wird der Name der Komponente zugewiesen.

Ausgabedatei:

Wählen Sie die Zieldatei aus, in der die Daten ausgegeben werden sollen, wenn **Code** mit MapForce **generiert** wird. Stellen Sie sicher, dass beim Generieren von Code eine unterschiedliche Eingabe- und Ausgabedatei angegeben wurde, da die Quelldatei sonst beim Ausführen des Codes überschrieben wird. Diese Option wird nur verwendet, wenn Code für Java, C++ und C# generiert wird oder wenn die mit dem Built-In-Ausführungsprozessor erstellte Ausgabedatei gespeichert wird.

Dateikodierung:

Dient zur Definition/Auswahl der Kodierung der Eingabe- und Ausgabedatei. Wurde im Feld Eingabedatei keine Codierung ausgewählt, wird automatisch das Standard-Encoding UTF-8 ausgewählt.

CSV Einstellungen - Feldbegrenzung:

Wählen Sie die Art der Feldbegrenzungen für die Textdatei aus (bei CSV-Dateien "Comma Separated Value" erfolgt die Trennung standardmäßig durch Kommas ","). Sie können auch ein benutzerdefiniertes Begrenzungszeichen im Feld **Benutzerdefiniert** eingeben.

Klicken Sie in das Feld "Benutzerdefiniert" und:

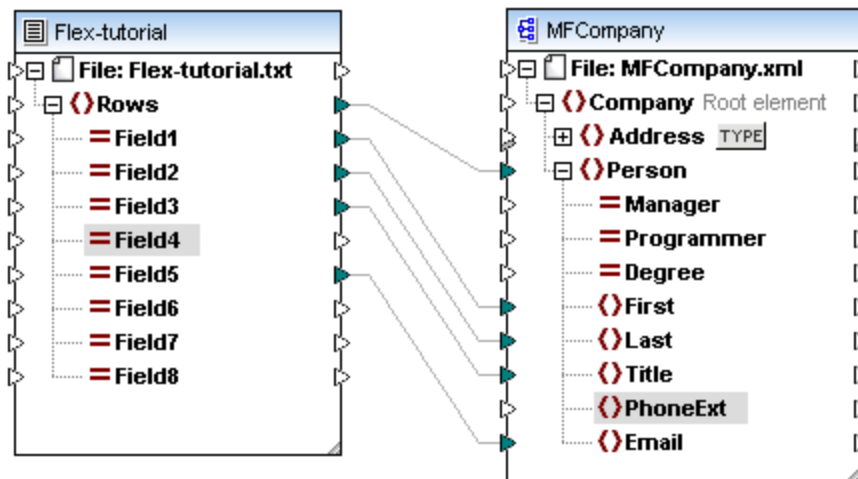
- drücken Sie auf der Tastatur eine Taste, um einen neuen Wert einzugeben oder
- doppelklicken Sie in das Feld "Benutzerdefiniert", um den aktuellen Wert zu markieren und drücken Sie eine andere Taste, um den Wert zu ändern

Erste Zeile enthält Feldnamen:

Definiert die **Werte** im ersten Feld der Textdatei als Spaltenüberschrift (wird im Vorschaufenster angezeigt). Die Spaltenüberschriften erscheinen in der Folge als Namen der Datenelemente, wenn die Textkomponente im Mapping angezeigt wird.

Leere Felder als nicht vorhanden behandeln

Damit können Sie definieren, dass für leere Felder in der Quelldatei in der Zieldatei kein entsprechendes leeres Datenelement (Element oder Attribut) erstellt wird.

**Aktiv:**

Eines der Felder der Quelldatei enthält nur Daten in Field1; Die Felder 2 bis 5 sind leer. Wenn diese Option aktiv ist, werden die Ziel-Datenelemente, die keine Daten aus der Quelldatei empfangen, in der Ausgabe nicht angezeigt (Zeile 39).

```

38  <Person>
39  ... <First>General outgassing pollutants</First>
40  </Person>
41  <Person>
42  <First>1100</First>
43  <Last>897</Last>

```

Inaktiv:

Für die leeren Felder der Quelldatei werden entsprechende Zielelemente in der Ausgabedatei erstellt, also in diesem Beispiel: die Elemente Last, Title und Email.

```

33  <Person>
34  <First>General outgassing pollutants</First>
35  <Last/>
36  <Title/>
37  <Email/>
38  </Person>
39  <Person>
40  <First>1100</First>
41  <Last>897</Last>

```

Bitte beachten Sie:

Die Trennzeichen für die leeren Felder müssen dennoch erhalten bleiben z.B. "General outgassing pollutants,,,,".

Textbegrenzung:

Aus Legacy-Systemen exportierte Textdateien enthalten manchmal Textwerte in Anführungszeichen, um sie von numerischen Werten zu unterscheiden.

Wählen Sie diese Option, wenn die Textdatei Strings enthält, die das aktuell definierte Feldbegrenzungszeichen enthalten. Dasselbe Begrenzungszeichen kann dann innerhalb eines Strings vorkommen, ohne dass dadurch die Aufteilung der Textdatei beeinträchtigt wird. So können Ihre Felder (Strings) z.B. ein Kommazeichen "," enthalten, doch verwenden Sie dieses Zeichen auch als Standard-CSV-Begrenzungszeichen.

Feld **anhängen**, Feld **einfügen**, Feld **löschen**:

Dient zum Anhängen, Einfügen oder Löschen von Feldern im Vorschaufenster, in dem die Struktur der CSV-Komponente definiert wird.

Nächste / Vorherige

Durch Klicken auf eine dieser Schaltflächen verschieben Sie die aktive Spalte im Vorschaufenster nach links oder rechts.

18.5 Mappen von Textdateien mit fester Länge (auf eine Datenbank)

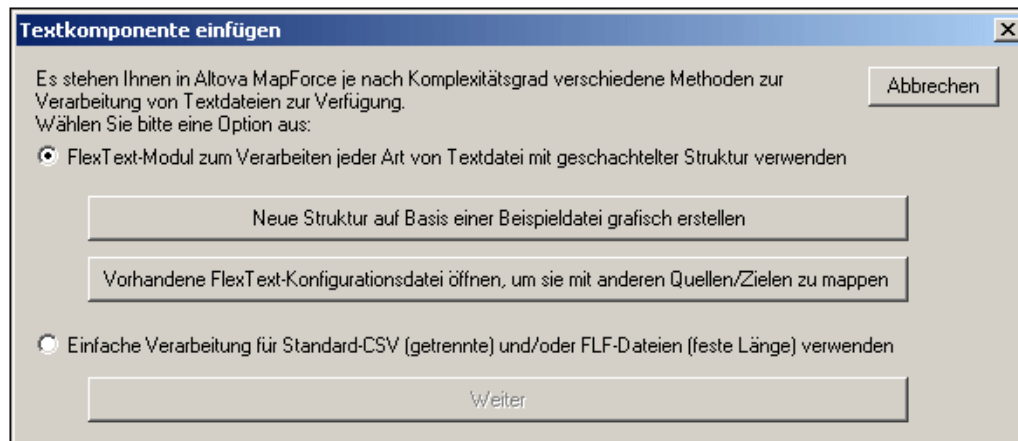
In diesem Beispiel wird eine einfache Textdatei auf eine MS Access Datenbank gemappt. Die Quelldatei besteht aus einem einzigen kontinuierlichen String ohne Wagenrücklauf- (carriage return) oder Zeilenvorschubzeichen (line feed). Sie finden alle im folgenden Beispiel verwendeten Dateien im Ordner ...**MapForceExamples\Tutorial**.

1. Wählen Sie die Menüoption **Einfügen | Textdatei** oder klicken Sie auf das Symbol



"Textdatei einfügen"

Daraufhin wird das Dialogfeld "Textkomponente einfügen" geöffnet.



Aktivieren Sie das Optionsfeld **Einfache Verarbeitung für ...** und klicken Sie auf **Fortfahren**.

Daraufhin wird das Dialogfeld "Text importieren / exportieren" geöffnet, wo Sie den Dateityp und die gewünschten Einstellungen festlegen können.

2. Klicken Sie auf die Schaltfläche **Eingabedatei** und wählen Sie die Datei **Altova-FLF.txt** aus.
Sie werden feststellen, dass die Datei aus einem einzigen String besteht und das Füllzeichen # enthält.

Komponenteneinstellungen

Komponentenname: Textdatei

OK
Abbrechen

Eingabe- / Ausgabedatei
Documents\Altova\MapForce2012\MapForceExamples\Tutorial\Altova-FLF.txt
Eingabedatei
Ausgabedatei

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- /Ausgabekodierung
Kodierungsname: Unicode UTF-8
Bytefolge: Little Endian ☐ Bytefolge-Markierung inkl.

CSV Einstellungen
Feldbegrenzung: ☐ Tab ☐ Semikolon ☒ Komma ☐ Leerz. ☐ Benutzerdef.
Textbegrenzung: ☐ Keine ☐ ' ☒ " ☒ CSV ☐ Feste Länge
☐ Erste Zeile enthält Feldnamen
☒ Leere Felder als nicht vorhanden behandeln

Field1
string
Vernon##Callaby##582v.callaby@nanonull.com##Office Manager#####Frank##Further##471f.further@na

Feld anhängen Feld einfügen Feld löschen << >>

3. Aktivieren Sie das Optionsfeld **Feste Länge** (unterhalb von CSV).
4. Deaktivieren Sie das Kontrollkästchen "**Datensatz-Trennzeichen als vorhanden annehmen**".

Field1
string
1
V
e
r
n

Zu diesem Zeitpunkt ändert sich die Vorschau. Angezeigt wird nun ein fixes Format bestehend aus:

- einen einzigen Feld mit dem Titel "Feld1"
 - das Format ist vom Typ "string" und die Feldlänge ist ein Zeichen (Vernon wird nun vertikal angezeigt!)
- Field 1 enthält nun zusätzliche Daten.

- Klicken Sie in das Feld, das die Feldlänge "1" enthält und ändern Sie den Wert in 8. Bestätigen Sie mit der Eingabetaste.

Field1
string
8
Vernon##
Callaby#
##582v.c
allaby@n

Feld anhängen Feld einfügen Feld löschen << >>

In der ersten Spalte, die nun als 8 Zeichen breit definiert ist, werden nun mehr Daten angezeigt.

- Klicken Sie auf die Schaltfläche **Feld anhängen**, um ein neues Feld anzuhängen und definieren Sie als Feldlänge 10 Zeichen.

Field1	Field2
string	string
8	10
Vernon##	Callaby###
582v.cal	laby@nanon

- Erzeugen Sie auf dieselbe Weise drei weitere Felder mit den folgenden Längen: 3, 25 und 25 Zeichen und ändern Sie die Spaltenüberschriften, um das Mappen zu erleichtern in: First, Last, Tel.-Ext, Email, Title. Die Vorschau sieht nun aus wie hier gezeigt:

Komponenteneinstellungen

Komponentenname: Textdatei

Eingabe- / Ausgabedatei: Documents\Altova\MapForce2012\MapForceExamples\Tutorial\Altova-FLF.txt

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- / Ausgabekodierung: Kodierungsname: Unicode UTF-8, Bytefolge: Little Endian, ☐ Bytefolge-Markierung inkl.

Felder mit fester Länge - Einstellungen: Füllzeichen: ☐ Leerz., ☐ Punkt, ☒ Benutzerdef. #, ☐ Datensatz-Trennzeichen als vorhanden annehmen, ☒ Leere Felder als nicht vorhanden behandeln

CSV ☒ Feste Länge

First	Last	Tel.-Ext	Email	Title
string	string	string	string	string
8	10	3	25	25
Vernon##	Callaby###	582	v.callaby@nanonull.com###	Office Manager####
Frank###	Further###	471	f.further@nanonull.com###	Accounts Receivable####

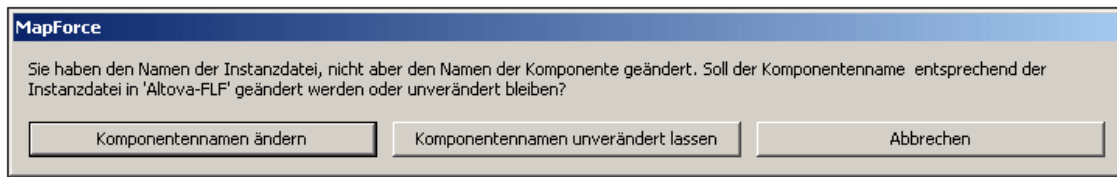
Feld anhängen, Feld einfügen, Feld löschen, <<, >>

8. Klicken Sie in der Gruppe "Felder mit fester Länge - Einstellungen" in das Textfeld "Benutzerdefiniert" und geben Sie dort als Füllzeichen die Raute (#) ein. Daraufhin wird dieses Füllzeichen aus der Eingabe-Textdatei entfernt.

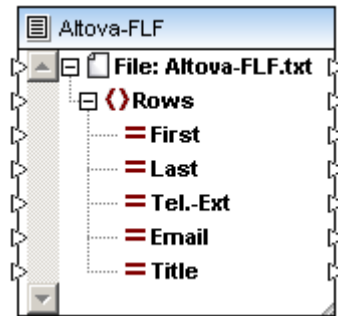
Felder mit fester Länge - Einstellungen: Füllzeichen: ☐ Leerz., ☐ Punkt, ☒ Benutzerdef. #, ☐ Datensatz-Trennzeichen als vorhanden annehmen, ☒ Leere Felder als nicht vorhanden behandeln

First	Last	Tel.-Ext
string	string	string
8	10	3
Vernon	Callaby	582
Frank	Further	471

9. Bestätigen Sie die Einstellungen mit OK. Daraufhin wird automatisch die unten gezeigte Eingabeaufforderung angezeigt. Alle Komponenten eines Mappings haben einen Komponentennamen. Der Standardname für eine Textdatei ist "Textdatei", wie oben gezeigt. Wenn Sie eine Textdatei als Input-Datei auswählen und dies mit OK bestätigen, werden Sie gefragt, ob Sie den Komponentennamen ebenfalls ändern möchten.



10. Klicken Sie auf die Schaltfläche "Komponentennamen ändern", um den Namen zu ändern.



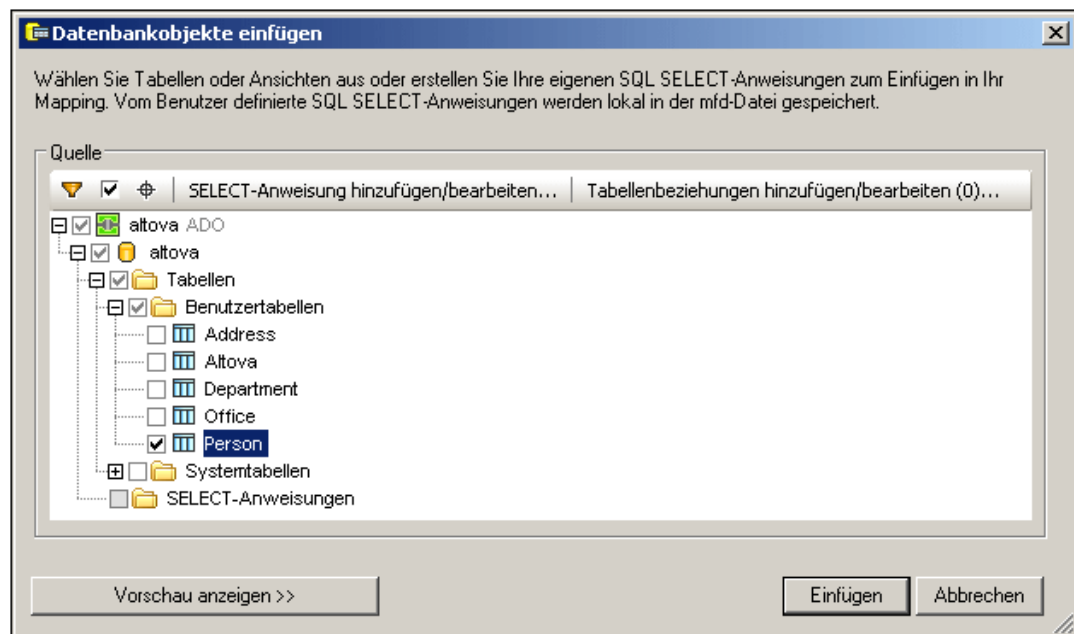
Die Textdateikomponente wird im Mapping-Fenster angezeigt. Sie können nun Daten von dieser und auf diese Komponente mappen.

Mappen von Textdateien auf eine Datenbank:

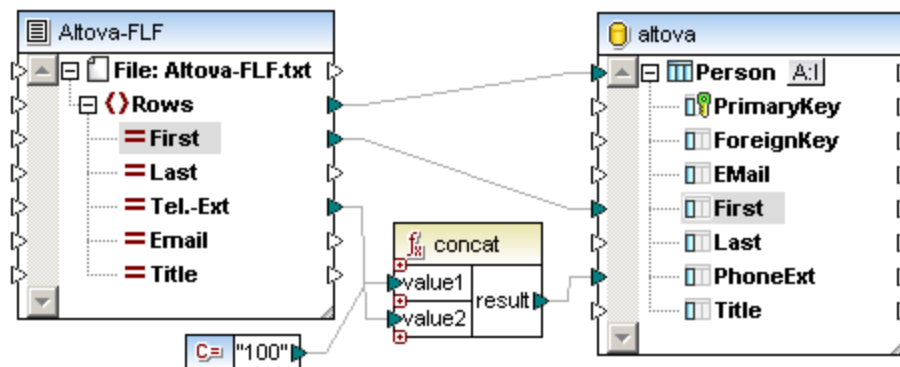
In diesem Abschnitt wird eine Textdatei mit fester Länge verwendet, um die Telefondurchwahleinträge in der Datenbank **altova.mdb** zu aktualisieren.

Person : Table				
	First	Last	PhoneExt	Titl
▶	Vernon	Callaby	582	Office Mana
	Frank	Further	471	Accounts F

1. Wählen Sie die Menüoption **Einfügen | Datenbank**, klicken Sie auf das Optionsfeld "Microsoft Access" und anschließend auf Weiter.
2. Wählen Sie die Datenbank **altova.mdb** aus dem Ordner **...MapForceExamples\Tutorial** und klicken Sie auf Weiter.
3. Wählen Sie die Tabelle **Person** durch Aktivieren des entsprechenden Kontrollkästchens in der Liste "Datenbank-Tabellen".

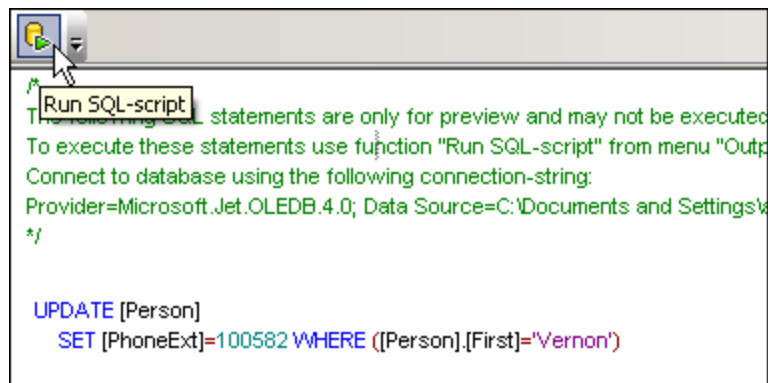


4. Klicken Sie auf die Schaltfläche **OK**, um die Datenbankkomponente zu erstellen.
5. Klicken Sie auf das Erweiterungssymbol, um den Inhalt der Tabelle zu sehen.
6. Ziehen Sie die **concat**-Funktion aus dem Bibliotheksfenster in das Mapping-Register.
7. Wählen Sie die Menüoption **Einfügen | Konstante**, aktivieren Sie das Optionsfeld "numerisch" und geben Sie als das neue Telefondurchwahlpräfix die Zahl 100 ein.
8. Erzeugen Sie das Mapping wie in der Abbildung unten gezeigt.



9. Klicken Sie auf das Tabellenaktionssymbol  neben dem Eintrag "Person".

10. Klicken Sie auf das Listenfeld "Aktion an Input-Daten" und wählen Sie den Eintrag "Aktualisieren, wenn...".
11. Klicken Sie auf das Listenfeld der Zeile "First", wählen Sie "equal" und bestätigen Sie mit OK.
- Die Daten der Tabelle "Person" werden nur dann aktualisiert, wenn der Vorname (First) im Feld der Quelle und dem der Datenbank identisch ist. Die Aktion, die durchgeführt wird, wenn dies "true" ist, ist im Mapping definiert. In diesem Fall wird der Telefondurchwahl die Zahl 100 vorangestellt und dieser Wert wird in das Feld PhoneExt der Tabelle "Person" gesetzt.
12. Klicken Sie auf die Schaltfläche "Ausgabe", um die Vorschau für die SQL-Anweisung zu generieren und anschließend auf die Schaltfläche "SQL-Skript ausführen" , um die SQL-Anweisungen auszuführen.



Die Telefondurchwahlfelder für alle Personen werden in der Datenbank aktualisiert.

First	Last	PhoneExt	Title
Vernon	Callaby	100582	Office Manager
Frank	Further	100471	Accounts Manager
Loby	Matise	100963	Accounting
Joe	Firstbread	100621	Marketing Manager
Susi	Sanna	100753	Art Director
Fred	Landis	100951	Program Manager

Record: 1 of 21

18.5.1 Optionen für Textdateien mit fester Länge

Rechtsklicken Sie auf die Textdatei-Komponente **Altova-FLF** und wählen Sie **Eigenschaften** aus, um das gleichnamige Dialogfeld zu öffnen.

Import-/Export-Optionen für Textdateien mit fester Länge:

Wenn Sie in diesem Dialogfeld Feldformate definieren, wird der Feldtyp der betreffenden Felder automatisch überprüft. Wenn die Eingabedaten nicht mit dem hier definierten Feldformat übereinstimmen, werden die Daten rot markiert.

Bitte beachten Sie:

Welche Feldtypen man für eine bestimmte Spalte auswählen kann, ist abhängig von den Standard-XML-Schema-Datentypen. So hat z.B. der Datumstyp die Form: JJJJ-MM-TT.

Komponenteneinstellungen

Komponentenname: Textdatei

OK
Abbrechen

Eingabe- / Ausgabedatei

Documents\Altova\MapForce2012\MapForceExamples\Tutorial\Altova-FLF.txt Eingabedatei

Ausgabedatei

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- / Ausgabekodierung

Kodierungsname: Unicode UTF-8

Bytefolge: Little Endian ☐ Bytefolge-Markierung inkl.

Felder mit fester Länge - Einstellungen

Füllzeichen

☐ Leerz. ☐ Punkt ☒ Benutzerdef.

☐ Datensatz-Trennzeichen als vorhanden annehmen

☒ Leere Felder als nicht vorhanden behandeln

☐ CSV ☒ Feste Länge

First	Last	Tel.-Ext	Email	Title
string	string	string	string	string
8	10	3	25	25
Vernon##	Callaby###	582	v.callaby@nanonull.com###	Office Manager####
Frank###	Further###	471	f.further@nanonull.com###	Accounts Receivable

Feld anhängen Feld einfügen Feld löschen << >>

Eingabedatei:

Wählen Sie die Textdatei, die als Quelldatei für diese Komponente verwendet werden soll.

Bitte beachten Sie:

Dieses Feld kann leer bleiben, wenn Sie die Textdateikomponente als **Zielkomponente** für ein Mapping verwenden. In diesem Fall wird als Standard-Encoding automatisch UTF-8 ausgewählt. Sie können den Feldtyp, die Feldnamen, die Formatierung usw. definieren, und die Einstellungen durch Klicken auf OK bestätigen, um eine Ziel-Textdatei zu erstellen.

Wenn Sie auf die Schaltfläche **Ausgabe** klicken, können Sie diese Textdatei mitsamt ihrem gemappten Inhalt durch Klicken auf das Symbol "Generierte Ausgabe speichern..."  **speichern**.

Wenn Sie in dieses Textfeld einen Namen eingeben (ohne Dateierweiterung), wird der Name der Komponente zugewiesen.

Ausgabedatei:

Wählen Sie die Zieldatei aus, in der die Daten ausgegeben werden sollen, wenn **Code** mit MapForce **generiert** wird. Stellen Sie sicher, dass beim Generieren von Code eine unterschiedliche Eingabe- und Ausgabedatei angegeben wurde, da die Quelldatei sonst beim Ausführen des Codes überschrieben wird. Diese Option wird nur verwendet, wenn Code für Java, C++ und C# generiert wird oder wenn die mit dem Built-In-Ausführungsprozessor erstellte Ausgabedatei gespeichert wird.

Dateikodierung:

Dient zur Definition/Auswahl der Zeichenkodierung der Eingabe- und der Ausgabe-Textdatei. Wird im Feld Eingabe- / Ausgabedatei nichts eingetragen, so wird automatisch das Standard-Encoding UTF-8 ausgewählt.

Füllzeichen

Mit Hilfe dieser Option können Sie definieren, welches Zeichen verwendet werden soll, um den Rest von Feldern mit fester Länge auszufüllen, wenn die eingehenden Daten kürzer sind als die jeweils definierte Feldlänge. Im Feld "Benutzerdef." können Sie Ihr eigenes Füllzeichen definieren.

Entfernen von Füllzeichen:

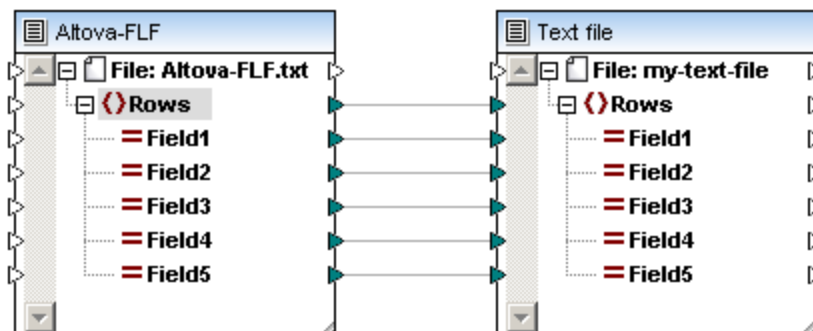
Wenn die eingehenden Daten bereits spezielle Füllzeichen enthalten, können Sie **dieses Zeichen** in das Feld "Benutzerdefiniert" eingeben, sodass diese Füllzeichen aus den eingehenden Daten entfernt werden.

Sie können auch ein benutzerdefiniertes Füllzeichen in das Feld **Benutzerdefiniert** eingeben. Klicken Sie in das Feld "Benutzerdefiniert" und:

- Drücken Sie eine Taste auf der Tastatur, um einen neuen Wert einzugeben oder
- doppelklicken Sie in das Feld "Benutzerdefiniert", um den aktuellen Wert zu markieren und drücken Sie eine andere Taste, um den Wert zu ändern.

Datensatz-Trennzeichen als vorhanden annehmen:

Wenn eine Textdatei mit fester Länge (ein einziger String) die Datenquelle für eine andere Textdatei mit fester Länge ist (Mappen von zwei Textdateien), dann werden bei Auswahl dieser Option in der **Zieldatei** neue Zeilen erstellt, nachdem die letzte Spalte des Ziels voll ist.



Im obigen Beispiel wird die Textdatei Altova-FLF auf die leere Ziel-Textdatei, my-text-file gemappt.

Komponenteneinstellungen

Komponentenname:

Eingabe- / Ausgabedatei

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- /Ausgabekodierung

Kodierungsname:

Bytefolge: ☐ Bytefolge-Markierung inkl.

Felder mit fester Länge - Einstellungen

Füllzeichen

☒ Leerz. ☐ Punkt ☐ Benutzerdef.

☐ Datensatz-Trennzeichen als vorhanden annehmen

☒ Leere Felder als nicht vorhanden behandeln

Field1	Field2	Field3	Field4	Fie
string	string	string	string	str
8	10	3	25	25

Bitte beachten Sie:

- Der Eintrag für die **Eingabedatei** bleibt leer, d.h. diese Textkomponente erhält nur Daten aus der gemappten Quellkomponente.
- Die definierten Feldlängen entsprechen den Feldlängen in der Datenquelle "Altova-FLF".
- In der Vorschau sind keine Daten zu sehen, da die Zielkomponente auf keiner existierenden Textdatei basiert.
- Wenn Sie auf die Schaltfläche "Ausgabe" klicken, werden die gemappten Daten angezeigt.

Kontrollkästchen "Datensatz-Trennzeichen als vorhanden annehmen"

- Ist die Option aktiv, wird nach der Summe der definierten Feldlängen ein neuer Datensatz angelegt, d.h. in diesem Fall beträgt die Summe der Zeichen in allen Feldern insgesamt 71. Für das 72. Zeichen wird ein neuer Datensatz erstellt.

```

1 Vernon##Callaby###582v.callaby@nanonull.com###Office Manager#####
2 Frank###Further###471f.further@nanonull.com###Accounts Receivable#####
3 Loby###Matise###963l.matise@nanonull.com###Accounting Manager#####
4 Joe###Firstbread621j.firstbread@nanonull.comMarketing Manager Europe#

```

- Ist die Option deaktiviert, werden die gemappten Daten mitsamt den definierten Füllzeichen als ein langer String dargestellt.

1	Vernon##Callaby###582v.callaby@nanonull.com###Office Manager#####Frank

Leere Felder als nicht vorhanden behandeln

Damit können Sie definieren, dass für leere Felder in der Quelldatei in der Zieldatei kein entsprechendes leeres Datenelement (Element oder Attribut) erstellt wird.

Aktiv:

Wenn diese Option aktiv ist, werden die Ziel-Datenelemente, die keine Daten aus der Quelldatei empfangen, in der Ausgabe nicht angezeigt.

Inaktiv:

Für die leeren Felder der Quelldatei werden entsprechende Zielelemente in der Ausgabedatei erstellt.

Feld anhängen, Feld einfügen, Feld löschen:

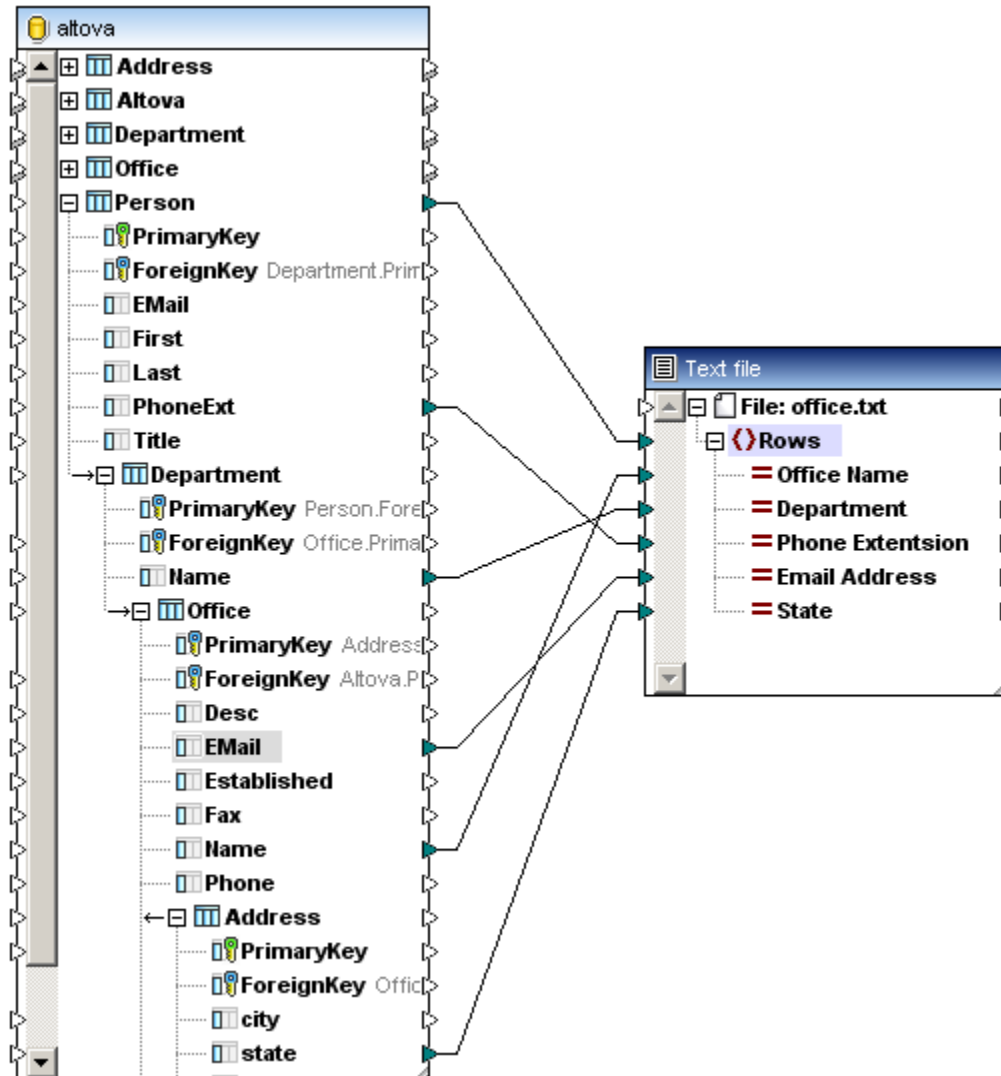
Dient zum Anhängen, Einfügen oder Löschen von Feldern im Vorschauenfenster, in dem die Struktur der Datei mit fester Länge definiert wird.

Nächste / Vorherige

Durch Klicken auf eine dieser Schaltflächen verschieben Sie die aktive Spalte im Vorschauenfenster nach links oder rechts.

18.6 Mappen von Datenbanken auf CSV/Text-Dateien

In diesem Beispiel wird eine einfache MS Access-Datenbank, altova.mdb, auf eine CSV-Datei gemappt. Die Datei altova.mdb liegt im Ordner ...\\MapForceExamples\\Tutorial\\.



Der Name der Datei Offices.txt, der in das Feld "Ausgabedatei" eingegeben wird, ist der Name, der automatisch vorgeschlagen wird, wenn Sie auf dem Register "Ausgabe" auf das Symbol "Generierte Ausgabe speichern" klicken.

Komponenteneinstellungen

Komponentenname: Textdatei

Eingabe- / Ausgabedatei:

offices.txt

☒ Alle Dateipfade relativ zur MFD-Datei speichern

Eingabe- / Ausgabekodierung:

Kodierungsname: Unicode UTF-8

Bytefolge: Little Endian ☐ Bytefolge-Markierung inkl.

CSV Einstellungen:

Feldbegrenzung: ☐ Tab ☐ Semikolon ☒ Komma ☐ Leerz. ☐ Benutzerdef.

Textbegrenzung: ☒ CSV ☐ Keine ☐ ' ' ☐ Feste Länge

☐ Erste Zeile enthält Feldnamen

☒ Leere Felder als nicht vorhanden behandeln

Office Name	Department	Phone Extension	Email Address	State
string	string	string	string	string

Feld anhängen Feld einfügen Feld löschen << >>

Klicken Sie auf das Symbol "Generierte Ausgabe speichern", um die Textdatei zu generieren/zu erzeugen.

	Office Name	Department	Phone Extension	Email Address	State
1	"Nanonull, Inc."	Administration	582	office@nanonull.com	CA
2	"Nanonull, Inc."	Administration	471	office@nanonull.com	CA
3	"Nanonull, Inc."	Administration	963	office@nanonull.com	CA
4	"Nanonull, Inc."	Marketing	621	office@nanonull.com	CA
5	"Nanonull, Inc."	Marketing	753	office@nanonull.com	CA

Kapitel 19

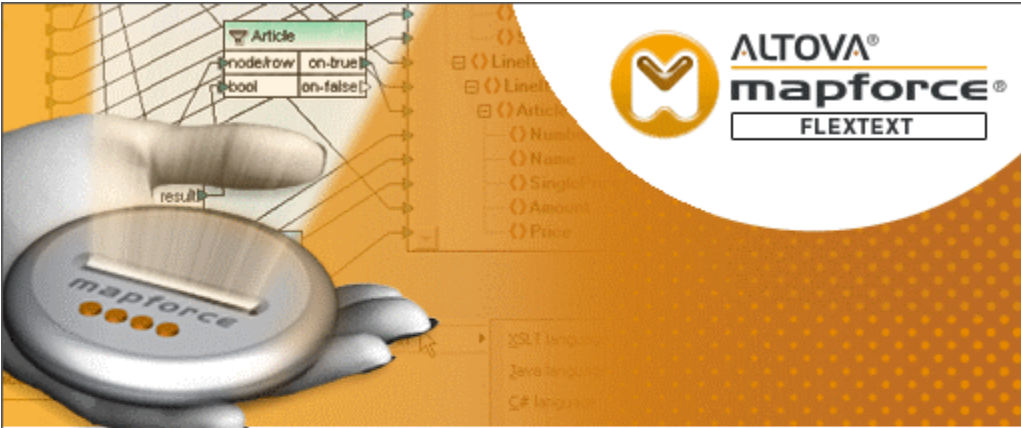
MapForce FlexText

19 MapForce FlexText

Das FlexText-Modul von MapForce 2012 gestattet die Verarbeitung und das komplexe Mappen von bestehenden Textdateien, so genannten Legacy-Textdateien.

Zuerst wird im FlexText-Modul eine FlexText-Vorlage zur Definition von Dateistruktur und -inhalt angelegt. Anschließend wird diese als Komponente in ein Mapping eingefügt, wo Sie nun entscheiden können, welche Elemente/Auswahl auf andere Zieldateien gemappt werden sollen.

Als Zieldateien kann jede der vielen Dateiarnten verwendet werden, die in MapForce unterstützt werden: Text-, XML-, Datenbank-Dateien oder EDI-Dateien.



Copyright © 1998–2012, Altova GmbH. All rights reserved. Use of this software is governed by and subject to an Altova software license agreement. XMLSpy, MapForce, StyleVision, SemanticWorks, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, AltovaXML, MissionKit, and ALTOVA as well as their logos are trademarks and/or registered trademarks of Altova GmbH. Patent pending.

XML, XSL, XHTML, and W3C are trademarks (registered in numerous countries) of the World Wide Web Consortium; marks of the W3C are registered and held by its host institutions, MIT, INRIA, and Keio. UNICODE and the Unicode Logo are trademarks of Unicode Inc. This software contains 3rd party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html

19.1 Übersicht

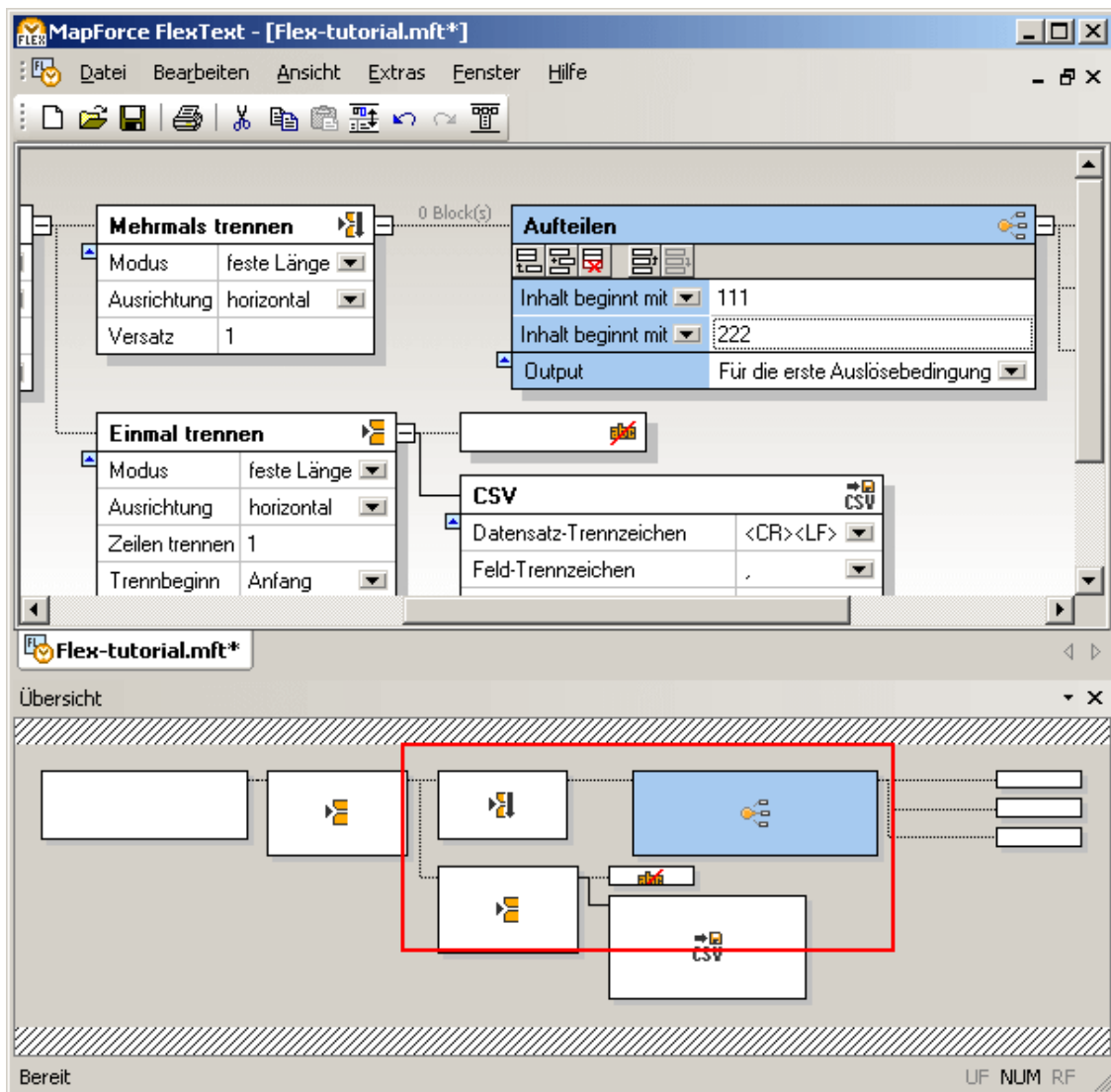
FlexText bietet eine grafische Benutzeroberfläche zum Mappen extrem komplexer Flat Files, die mehrere Trennzeichen enthalten, geschachtelter zeileninterner Strukturen und anderer komplexer Strukturen in MapForce, sodass Sie diese für Ihre Codegenerierungszwecke einsetzen können.

FlexText bietet eine Vorlage, die anschließend in MapForce geladen wird, über die die einzelnen Elemente auf jede Art von Zielkomponente gemappt werden können. Die Vorlage basiert auf einer Textdatei, die in MapForce zur Verfügung gestellt/geöffnet wird. Sie können dadurch dieselbe Vorlage für mehrere Textdateien in mehreren Mappings verwenden.

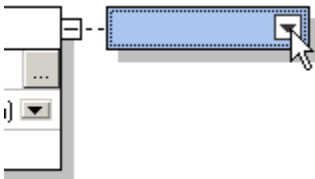
In FlexText können Sie die Struktur der Flat File interaktiv definieren und sehen im Beispieltextbereich sofort, wie sich Ihre Bearbeitungen auswirken.

FlexText besteht aus drei Hauptfenstern: Design, Übersicht und dem Beispieltext-Fenster.

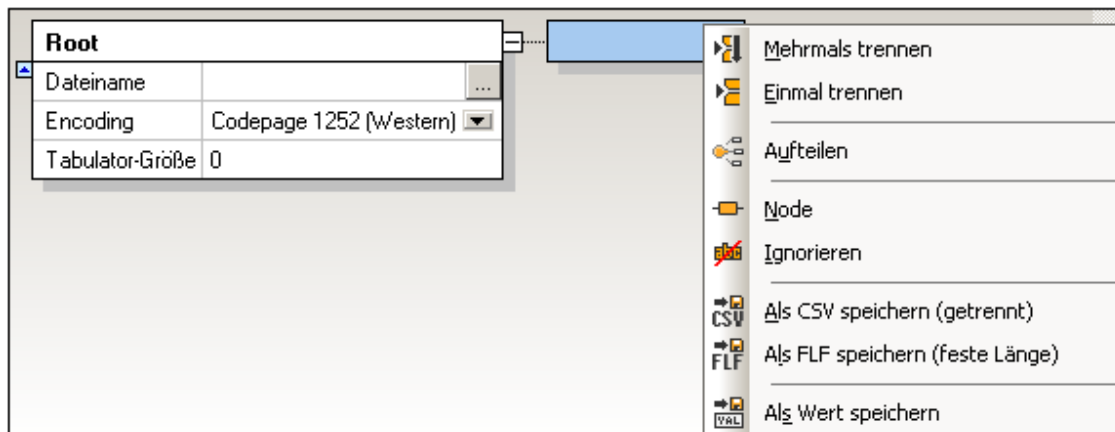
- Das Design-Fenster enthält Textfragment-Container mit Standardnamen, die deren Funktion beschreiben, z.B. "Mehrmals trennen".
- In den einzelnen Containern bzw. im Beispieltextbereich wird der Inhalt des derzeit aktiven Containers angezeigt.
- Das Übersichtsfenster zeigt eine allgemeine Übersicht über alle Container im Design-Fenster. Mit Hilfe des roten Rechtecks können Sie im Design-Fenster navigieren.



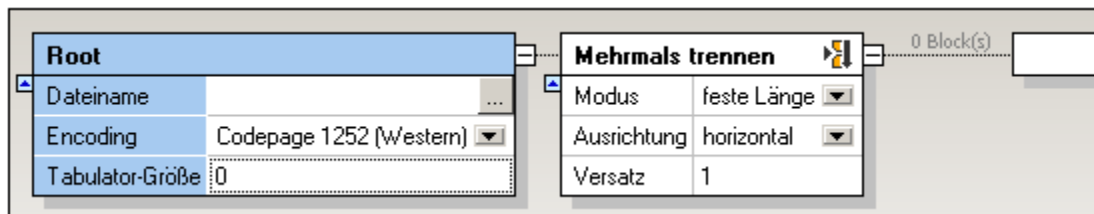
Die Container verfügen über anklickbare Symbole, mit Hilfe derer Sie die Art und den Inhalt des Containers definieren können.



Wenn Sie auf das Container-Symbol klicken, wird ein Popup-Menü geöffnet, aus dem Sie den Container-Typ auswählen können.



Jede der gewählten Optionen definiert die Funktion und den Inhalt des Containers und bietet Ihnen weitere Optionen zur genaueren Definition des Inhalts, der der Zielkomponente in MapForce zur Verfügung gestellt werden soll.



Nach Auswahl der Option:

- ändert der Container sein Aussehen, den Typ und das Symbol, so erscheint z.B. in der Titelleiste "Mehrmals trennen"
- werden Standardoptionen angezeigt z.B.: Modus=feste Länge, Ausrichtung=horizontal und Versatz=1.
- wird automatisch ein neuer Container an den aktuellen angehängt.
- Durch Klicken auf die Symbole "erweitern"  / "reduzieren"  können Sie Container-Bereiche ein- und ausblenden.
- Wenn Sie die Umschalttaste drücken, können Sie eine Gruppe von Containern reduziert anzeigen. Wenn Sie die Umschalttaste drücken erscheint ein Erweiterungssymbol . Wenn Sie auf dieses Symbol klicken, wird der Abschnitt der Containerstruktur, der sich rechts von der Struktur befindet, auf die geklickt wurde, reduziert angezeigt.

Bei Auswahl des Symbols "Node-Text in Design-Ansicht"  wird der Inhalt des aktiven Containers im Beispieltextrbereich angezeigt.

Bei Auswahl des Symbols "Node-Text für nicht ausgewählte automatisch reduzieren"  wird der Inhalt des aktiven Containers angezeigt. Alle anderen Container, die Inhalt enthalten, werden reduziert angezeigt.

19.2 FlexText Tutorial

Im Tutorial wird erläutert, wie Sie die wichtigsten Funktionen von FlexText verwenden, um eine Textdatei zu verarbeiten und das Resultat auf verschiedene Arten in MapForce zu mappen.

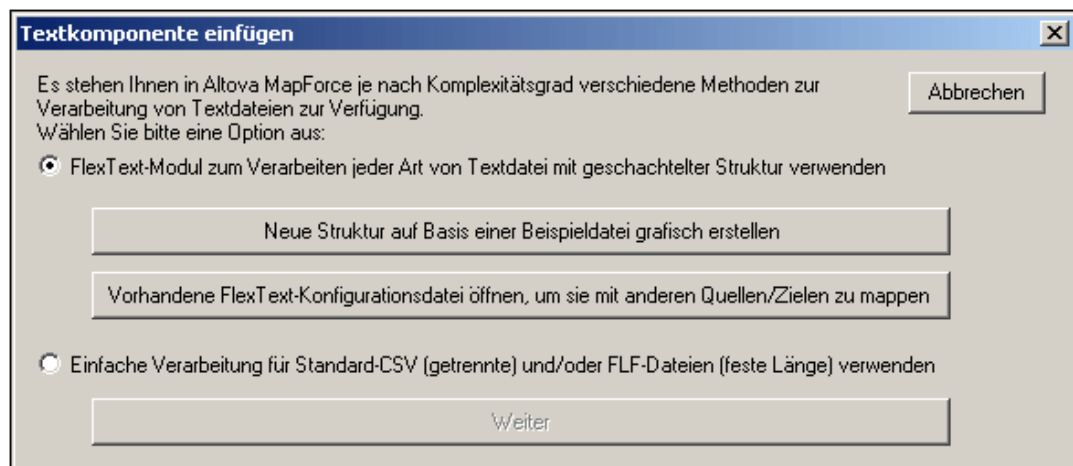
Im Beispiel wird die Datei **Flex-tutorial.txt** aus dem Ordner **...\\MapForceExamples\\Tutorial** verwendet. Sie hat das folgende Format:

```
111,332.1,22537.7,5,Container ship,Mega,  
111,A1579227,10,3,400,Microtome,  
111,B152427,7,6,1200,Miscellaneous,  
222,978.4,7563.1,69,Air freight,Mini,  
222,ZZAW561,10,5,10000,Gas Chromatograph,,  
  
General outgassing pollutants  
1100,897,22.1,716235,LOX  
1110,9832,22991.30,002,NOX  
1120,1213,33.01,008,SOX
```

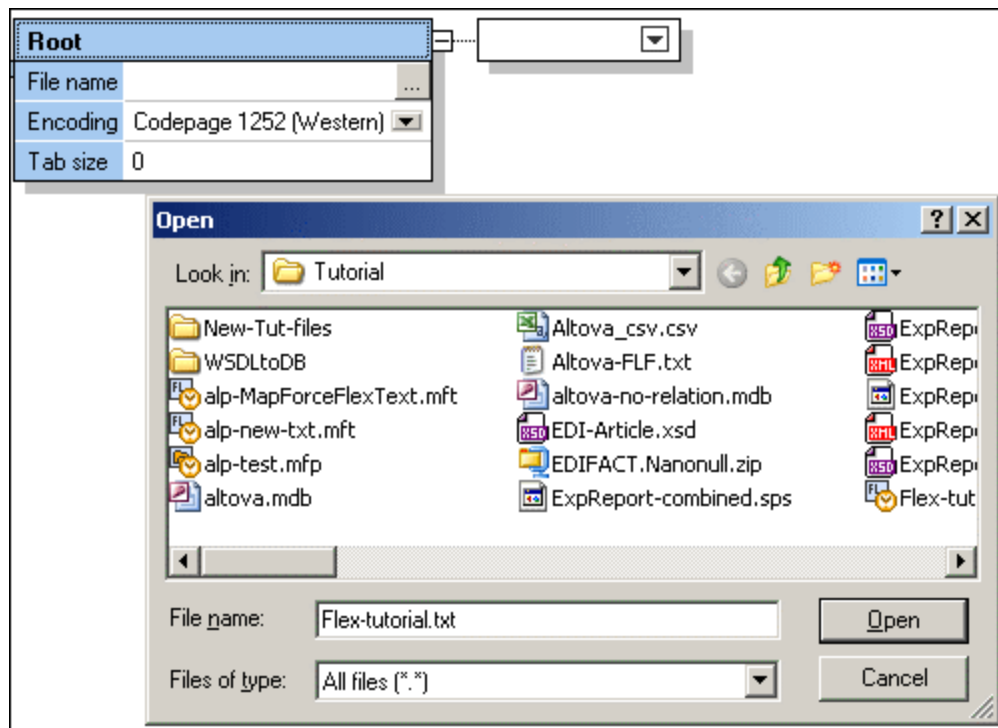
Ziel dieses Tutorials:

- die Datensätze mit den Schlüsseln, die die Zahlen 111 und 222 enthalten, herauszutrennen und in Elementen abzulegen, die separat gemappt werden können,
- die reinen Textstellen zu entfernen,
- eine CSV-Datei mit den verbleibenden Datensätzen zu erstellen.

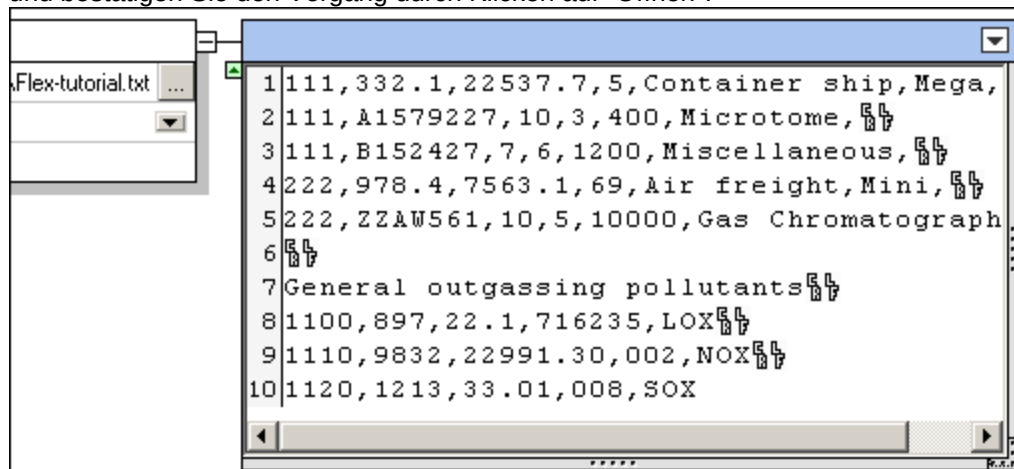
1. Starten Sie MapForce und öffnen Sie eine neue Mapping-Datei.
2. Wählen Sie **Einfügen | Textdatei** oder klicken Sie auf das Symbol "Textdatei einfügen" .
3. Klicken Sie auf die Schaltfläche **"Neue Struktur auf Basis einer Beispieldatei grafisch erstellen"**.



4. Geben Sie einen Namen für Ihre FlexText Vorlage ein und klicken Sie auf "Speichern" (z.B. Flex-tutorial.mft) um fortzufahren. Zusammen mit dem Dialogfeld "Öffnen" wird ein leeres Design angezeigt.



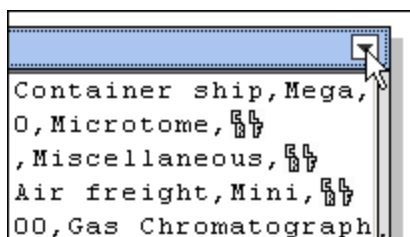
5. Wählen Sie im Ordner **...MapForceExamples\Tutorial** die Datei **Flex-tutorial.txt** aus und bestätigen Sie den Vorgang durch Klicken auf "Öffnen".



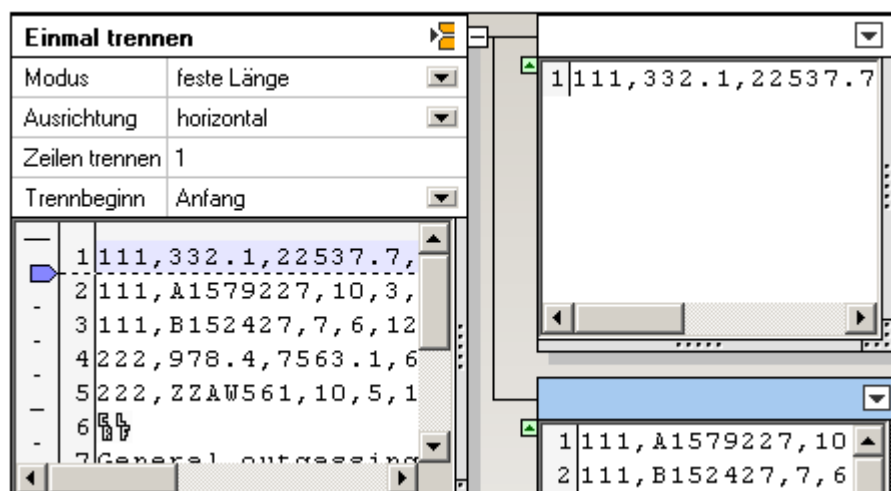
Sie sehen nun den Inhalt der Textdatei.

Wenn Sie auf das Symbol "Node-Text in Design-Ansicht"  klicken, wird der Inhalt des aktiven Containers im Beispieltextbereich angezeigt.

Wenn Sie das Symbol "Nicht ausgewählten Node-Text automatisch reduzieren"  klicken, wird der Inhalt im aktiven Container angezeigt. Alle anderen Container, die Inhalt enthalten, werden reduziert angezeigt.



6. Klicken Sie auf das Container-Symbol in der rechten oberen Ecke und wählen Sie aus dem Popup-Menü den Befehl **Einmal trennen**. Neben dem "Einmal trennen"-Container werden zwei neue Container angezeigt. Nähere Informationen zur Bedingung "Einmal trennen" finden Sie unter: [Einmal trennen](#).



Sie sehen die Standardeinstellungen des "Einmal trennen"-Containers: feste Länge, horizontal und Zeilen trennen=1.

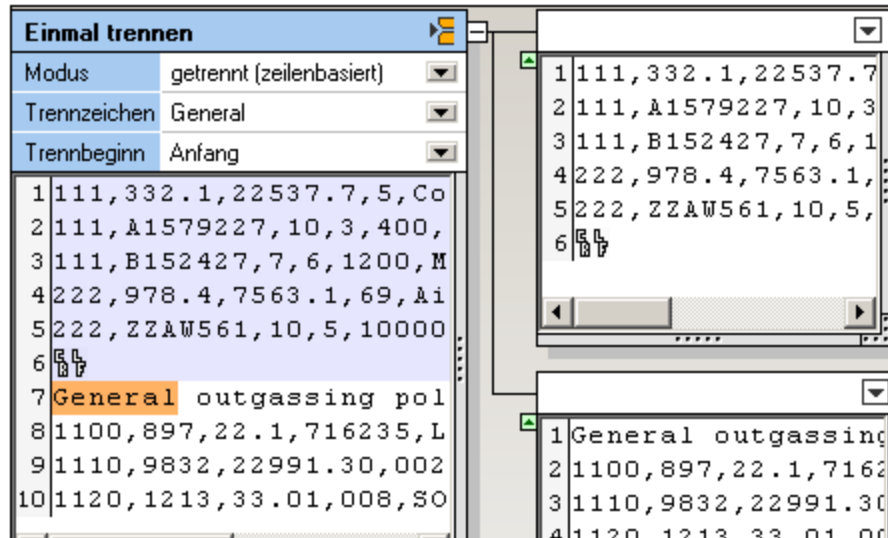
Außerdem sehen Sie auch das Ergebnis dieser Standardeinstellung:

- Der obere Container enthält die erste Zeile der Textdatei, die im Container "**Einmal trennen**" markiert erscheint.
- Der untere Container enthält den Rest der Textdatei.

19.3 Erstellen von Trennbedingungen

FlexText gestattet Ihnen, so genannte "Trennbedingungen" zu definieren, mit Hilfe derer Sie Textfragmente auf verschiedene Arten in Abschnitte aufteilen können.

1. Klicken Sie auf das Listenfeld **Modus** und wählen Sie "getrennt (zeilenbasiert)".
2. Doppelklicken Sie auf das Feld "Trennzeichen" und geben Sie "General" ein.

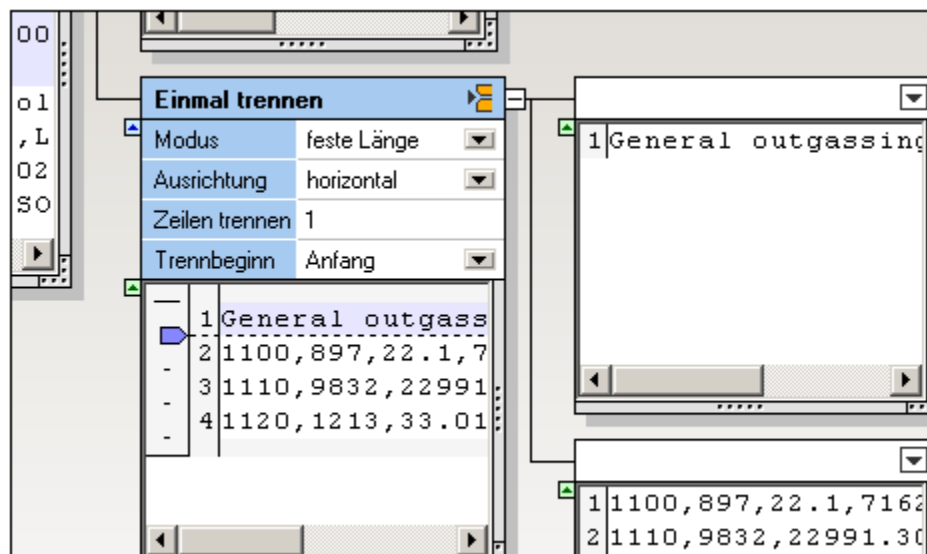


Die Textfragmente in den jeweiligen Containern haben sich nun geändert.

Durch Eingabe von "General" und Verwendung der Option "getrennt (zeilenbasiert)", können Sie den Textabschnitt, der die Zeichenfolge "General" enthält, abtrennen und in den unteren Container platzieren. Das Textfragment bis zum Trennzeichen wird in den oberen Container platziert.

Wir möchten nun den unteren Container bearbeiten, um eine CSV-Datei zu erzeugen, die die Datensätze mit 1100 und darüber enthält.

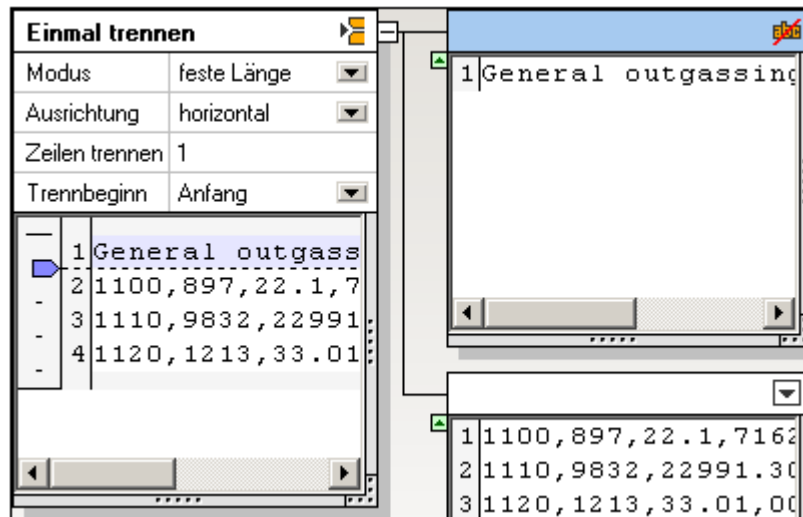
3. Klicken Sie auf den unteren Container und ändern Sie ihn in **Einmal trennen**.



Daraufhin werden zwei neue Container erstellt. Die Standardeinstellungen bleiben unverändert, da wir nur die erste Zeile dieses Textfragments abtrennen und ignorieren

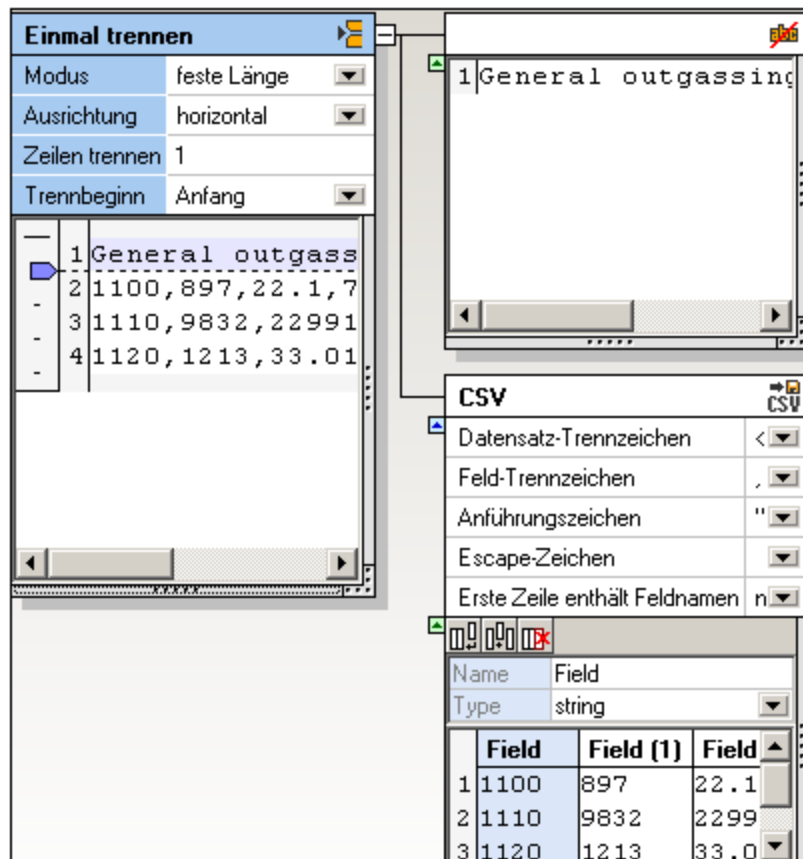
möchten. Aus dem restlichen Fragment im unteren Container wird eine CSV-Datei erzeugt.

4. Klicken Sie auf den oberen Container und ändern Sie ihn in **Ignorieren**.



Das Textfragment und somit Mapping-Element dieses Containers steht nun für das Mappen in MapForce nicht mehr zur Verfügung.

5. Klicken Sie auf den unteren Container und ändern Sie ihn in **Als CSV speichern**.



Der Container zeigt nun das Textfragment in Tabellenform an. Die Standardeinstellungen können beibehalten werden.

Konfigurieren der CSV-Datei:

Wenn Sie die Feldnamen ändern möchten, klicken Sie in der Tabelle in das Feld und ändern Sie den Eintrag im Feld **Name**. Des Weiteren können in diesem Container Spalten angehängt, eingefügt und gelöscht werden. Nähere Informationen dazu finden Sie unter "[Als CSV speichern](#)".

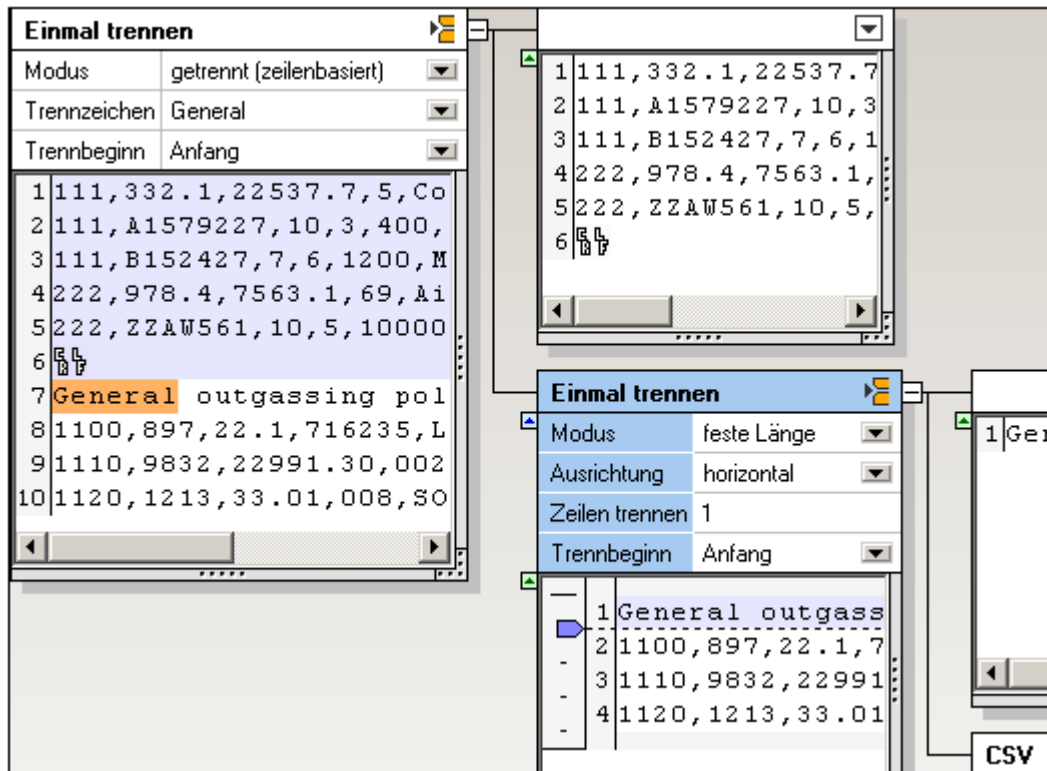
Wir können nun mit der Definition des restlichen Textfragments fortfahren.

19.4 Definieren mehrerer Bedingungen pro Container/Fragment

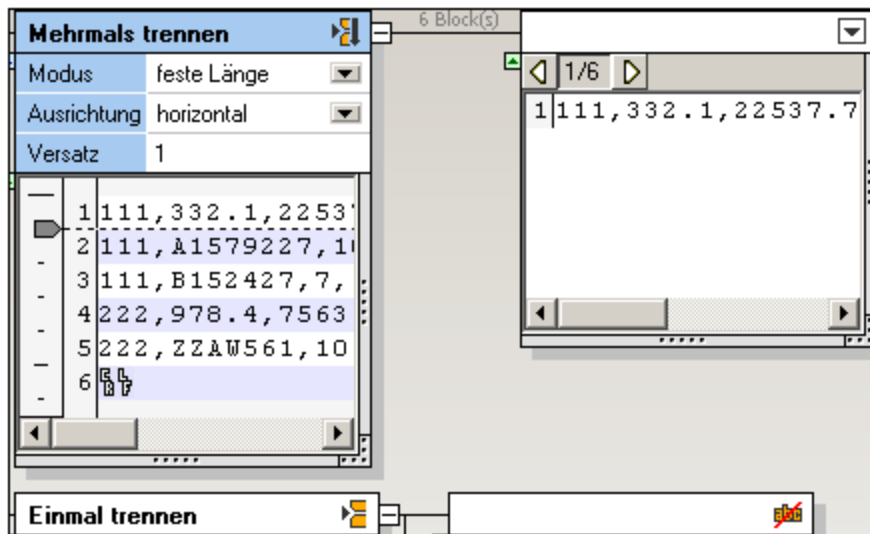
Sie können in FlexText mit Hilfe des Containers "Aufteilen" mehrere Bedingungen pro Textfragment definieren. Jede Bedingung, die Sie definieren, erhält automatisch einen dazugehörigen Container.

Bis zu diesem Zeitpunkt haben wir in diesem Tutorial das untere Textfragment des ersten **Einmal trennen**-Containers definiert:

- Mit einem "Einmal trennen"-Container wird die erste Zeile abgetrennt und in einen "Ignorieren"-Container platziert.
- Das restliche Segment wird als CSV-Datei definiert/gespeichert.



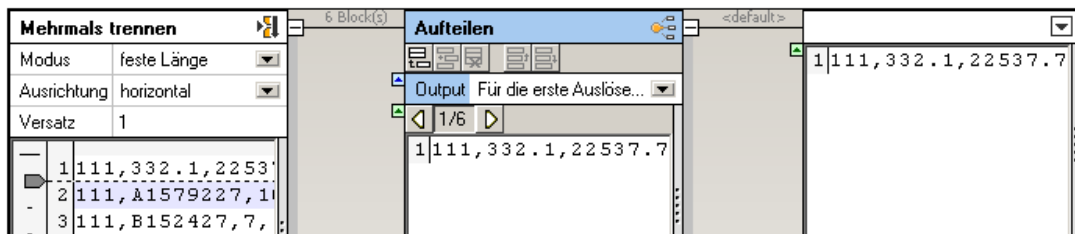
1. Klicken Sie auf das Symbol des oberen Containers und ändern Sie es in **Mehrmals trennen**.



Wir benötigen hier die Standardeinstellungen. Das Textfragment wird in mehrere einzeilige Textblöcke aufgeteilt. Im dazugehörigen Container sehen Sie ein Vorschau der einzelnen Textblöcke.

Wenn Sie auf das Symbol "Nächster Textblock"  klicken, können Sie der Reihe nach alle 6 Textfragmente anzeigen.

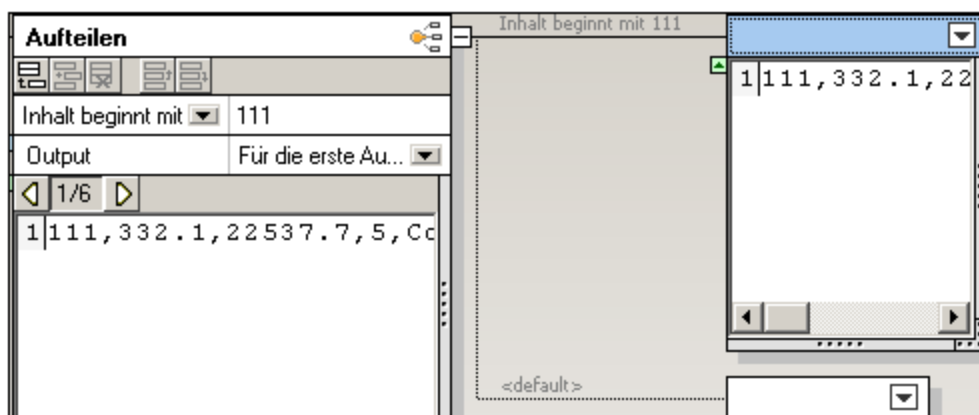
2. Klicken Sie auf den neuen Container und ändern Sie ihn in **Aufteilen**.



In der Abbildung oben sehen Sie den Anfangszustand des "Aufteilen"-Containers.

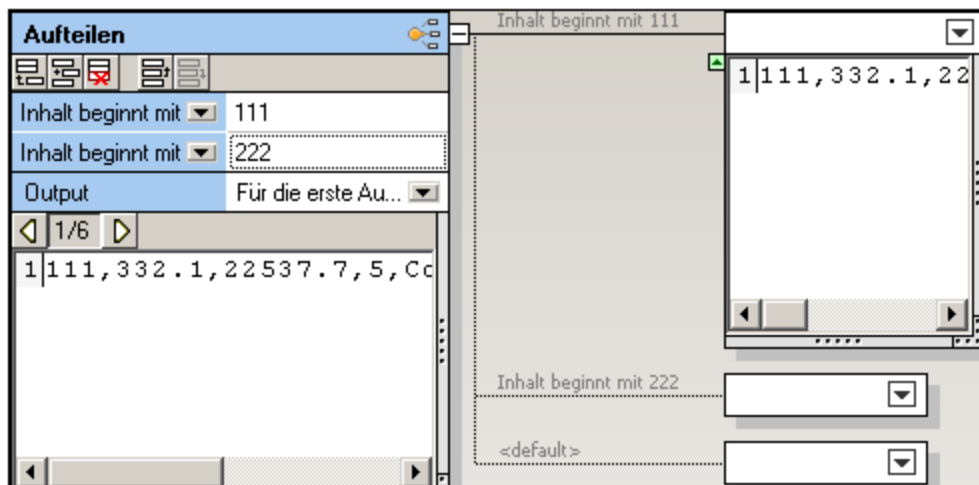
- Es wurde ein dazugehöriger Container "Default" hinzugefügt.
- Der Inhalt des ersten Datensatzes 1/6 wird im Standardcontainer angezeigt.

3. Klicken Sie in der Titelleiste "Aufteilen" auf das Symbol "Bedingung anhängen" , um eine neue Bedingung hinzuzufügen.
4. Doppelklicken Sie in das Feld "Inhalt beginnt mit" und geben Sie die Zahl 111ein.



Damit wird die erste Bedingung definiert. Oberhalb des "Standardcontainers" wurde ein dazugehöriger Container (Inhalt beginnt mit 111) hinzugefügt.

5. Klicken Sie nochmals auf das "Anhängen"-Symbol und geben Sie in das Feld "Inhalt beginnt mit" die Zahl "222" ein.

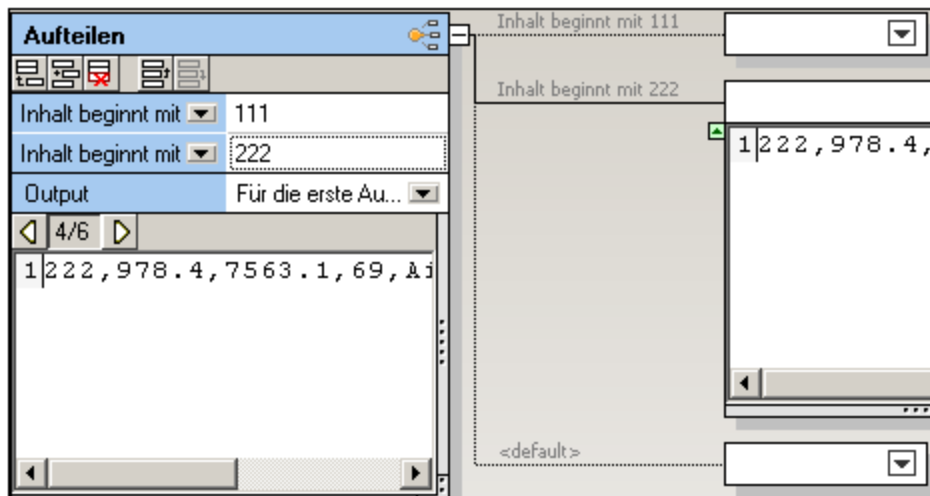


Es wurde ein dritter Container hinzugefügt (Inhalt beginnt mit 222).

Bitte beachten Sie:

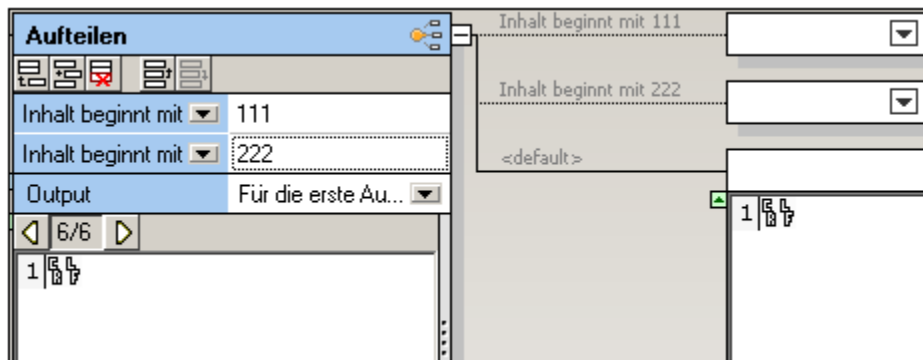
Wenn Sie auf das Listenfeld "Inhalt beginnt mit" klicken, können Sie die Option "Inhalt enthält" auswählen und dadurch eine Zeichenfolge definieren, die an einer beliebigen Stelle im Textfragment vorkommen kann.

6. Klicken Sie mehrmals hintereinander auf das Symbol "Nächster Textblock" , um die Auswirkung zu sehen.



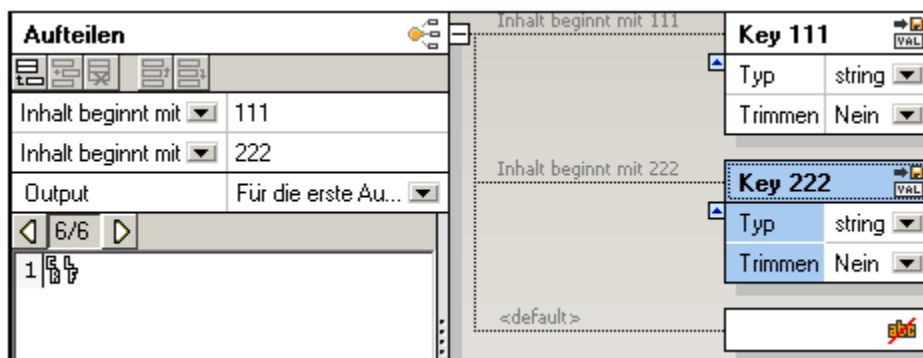
Sobald Sie Datensatz 4 von 6 erreichen, wird der Container 222 geöffnet und dessen Inhalt angezeigt.

7. Klicken Sie weiter, bis Sie Datensatz 6 von 6 erreichen. Im Standardcontainer wird ein einziges CR / LF-Zeichen angezeigt.



Wenn ein Datenfragment im aktuellen Block eine Bedingung erfüllt, so werden **alle Daten** dieses Blocks an den dazugehörigen Container übergeben. Die Daten werden nicht getrennt, sondern nur in den dazugehörigen Container platziert bzw. in den Standardcontainer, wenn keine der definierten Bedingungen erfüllt wird.

8. Klicken Sie auf die beiden ersten Container und ändern Sie sie in **Als Wert speichern**.
9. Doppelklicken Sie auf den Text "Speichern" und fügen Sie beschreibenden Text hinzu, z.B. Key 111 und Key 222.

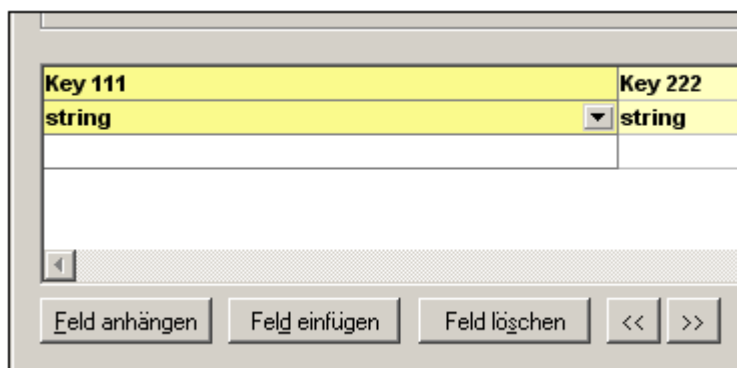


10. Speichern Sie die FlexText-Vorlage, z. B. als Flex-Tutorial.mft.

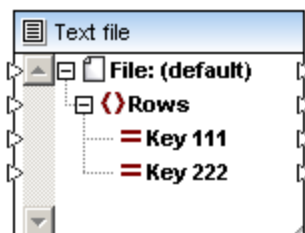
19.5 Verwendung von FlexText-Vorlagen in MapForce

Erstellen der Zielkomponenten für die Verwendung mit der FlexText-Quelle:

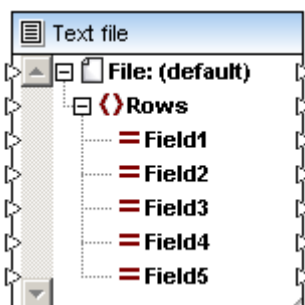
1. Wählen Sie den Befehl **Einfügen | Textdatei** oder klicken Sie auf das Symbol .
"Textdatei" .
2. Klicken Sie auf das Optionsfeld **"Einfache Verarbeitung...verwenden"** und klicken Sie auf **Weiter**.
Daraufhin wird das Dialogfeld "Komponenteneinstellungen" geöffnet.
3. Klicken Sie auf die Schaltfläche **Feld anhängen**, um ein neues Feld anzuhängen.
4. Doppelklicken Sie auf den Feldnamen Field1 und ändern Sie ihn in **Key 111**.



5. Gehen Sie mit dem anderen Feld ebenso vor, nennen Sie es **Key 222** und klicken Sie zur Bestätigung auf OK.
Es wurde nun eine Textkomponente mit zwei Feldern erstellt.

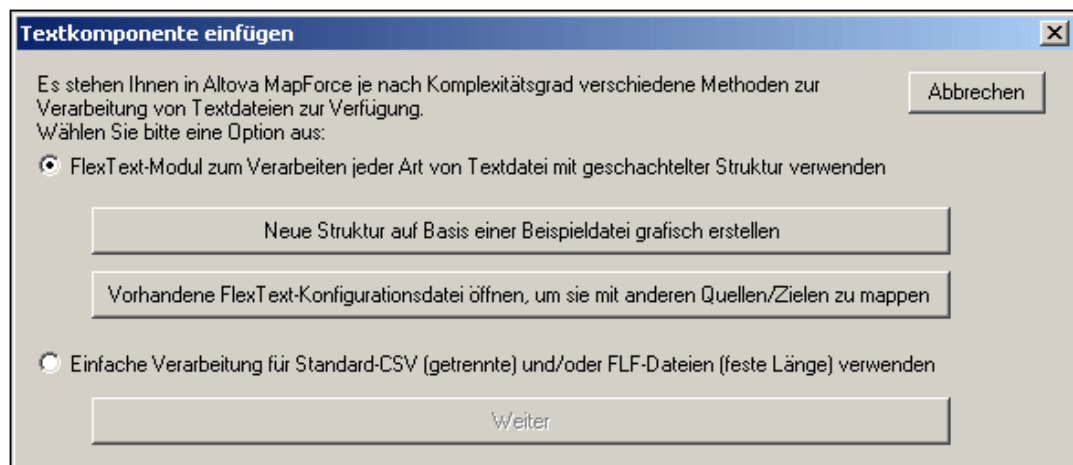


6. Erstellen Sie auf dieselbe Art eine zweite Textkomponente, die aus fünf Feldern besteht.

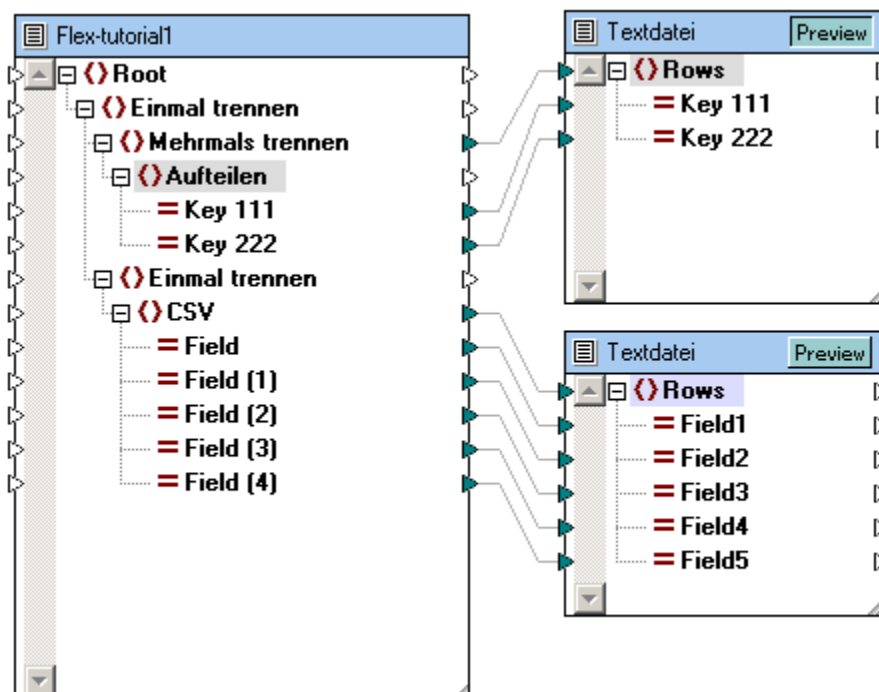


Verwendung der FlexText-Vorlage in MapForce:

1. Starten Sie MapForce bzw. wechseln Sie zurück zu MapForce und wählen Sie **Einfügen | Textdatei**.



2. Klicken Sie auf die Schaltfläche **Vorhandene FlexText-Datei öffnen,...**, und wählen Sie die zuvor definierte FlexText-Vorlage (z.B. Flex-tutorial.mft) aus. Die Struktur der MapForce-Komponente spiegelt die der Container in der Design-Ansicht in FlexText wider.



3. Mappen Sie die verschiedenen Elemente auf die zuvor definierten Zielkomponenten und klicken Sie auf das Register "Ausgabe-Vorschau", um eine Vorschau der Ergebnisse zu sehen.

Mapping-Vorschau der oberen Textkomponente:

1	"111,332.1,22537.7,5,Container ship,Mega,
2	",
3	"111,A1579227,10,3,400,Microtome,
4	",
5	"111,B152427,7,6,1200,Miscellaneous,
6	",
7	,"222,978.4,7563.1,69,Air freight,Mini,
8	"
9	,"222,ZZAW561,10,5,10000,Gas Chromatograph,,
10	"
11	

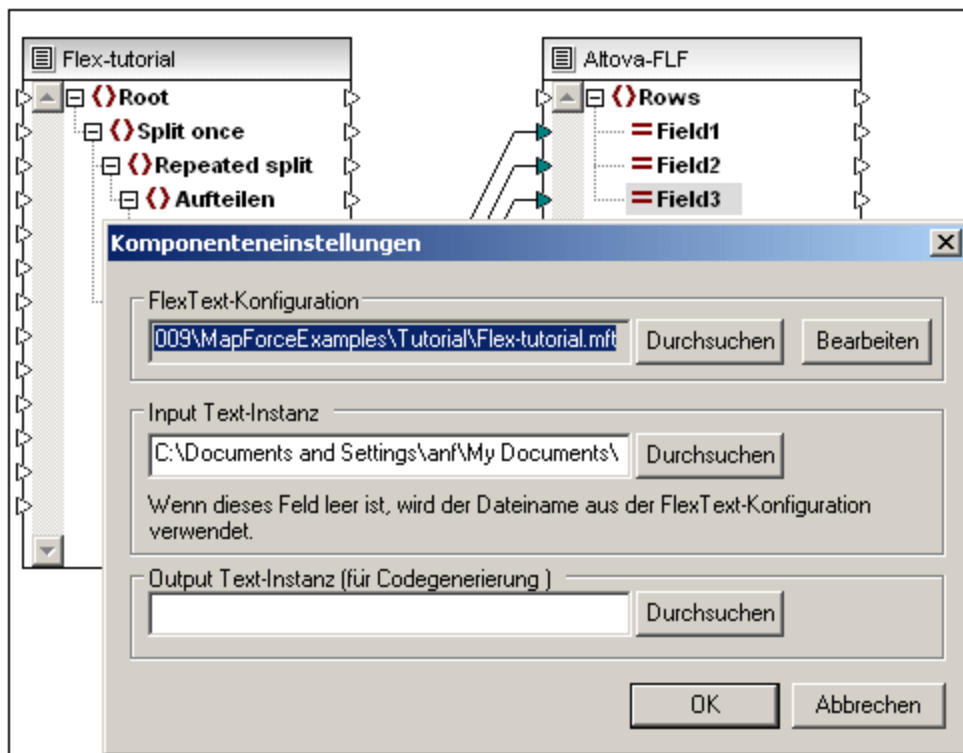
Mapping-Vorschau der unteren Textkomponente:

1	1100,897,22.1,716235,LOX
2	1110,9832,22991.30,002,NOX
3	1120,1213,33.01,008,SOX
4	

19.5.1 FlexText Datenquellen

Wenn Sie auf eine FlexText Komponente doppelklicken, wird das Dialogfeld **Komponenteneinstellungen** geöffnet, in dem Sie die verschiedenen Input-und Output-Dateien definieren können.

Das Feld **FlexText Konfiguration** enthält die bei der Erstellung der Komponente definierte Datenquelle.



Wenn im Feld **Input Text-Instanz** ein Eintrag vorhanden ist, wird diese Datenquelle anstelle der Quelle im Feld FlexText Konfiguration verwendet.

Der Eintrag im Feld **Output Text-Instanz** wird nur bei der Codegenerierung verwendet. Geben Sie beim Generieren von Code einen Namen ein, da FlexText automatisch einen Dateinamen generiert. Wenn Sie einen vollständigen Pfad eingeben, können Sie außerdem auch das Zielverzeichnis definieren, z.B. c:\myfiles\sequence.txt.

19.6 Verwenden von FlexText als Zielkomponente

Hauptsächlich werden FlexText-Komponenten verwendet, um Textdateien zu reduzieren oder Abschnitte daraus herauszutrennen und die relevanten Einträge der FlexText Komponente auf andere Zielkomponenten zu mappen. FlexText kann jedoch auch als Zielkomponente verwendet werden, um separate Dateien zu einer einzigen Datei zusammenzusetzen bzw. neu zusammenzusetzen. Oft sind jedoch einige Versuche erforderlich, um das gewünschte Ergebnis zu erzielen.

Wenn Sie eine FlexText Komponente als **Zielkomponente** verwenden, werden die zuvor darin definierten Operationen umgekehrt. Anstatt eine Datei in verschiedene Unterabschnitte aufzuteilen, wird sie zu einer einzigen Datei zusammengesetzt bzw. erneut zusammengesetzt.

Im Allgemeinen werden die einzelnen in der FlexText Vorlage definierten Operationen (von unten nach oben) umgekehrt, wenn sie als Ziel verwendet werden:

- eine Trennung wird zusammengeführt, so wird z.B. ein Mapping auf wiederholt durch "," getrennte Dateien zu einer Zusammenführung von Datenelementen, die durch "," getrennt sind.
- aus Speichern wird Laden
- aus "Aufteilen" wird "wähle die erste Übereinstimmung"

FlexText Komponente als Zielkomponente

Sobald eine Verbindung zwischen einer Datenquell-Komponente und einem der Input-Datenelemente einer FlexText Komponente hergestellt wurde, wird die Datenquelle der FlexText Komponente ignoriert. Vorrang haben nun die Daten aus der neu gemappten Quellkomponente.

Anmerkungen:

- Wenn Text auf einen "Speichern als..."-Container gemappt wird (Als CSV und FLF speichern), wird das Trennzeichen beibehalten. Es kann jedoch vorkommen, dass Text abgeschnitten wird, wenn in einem Node oberhalb des "Speichern als..." Node eine Trennung mit fester Länge vorkommt.
- Bei Trennungen mit fester Breite wird der linke/obere Abschnitt auf die vordefinierte Länge abgeschnitten, wenn Trennbeginn=Anfang, bzw. der rechte/untere Abschnitt bei Trennbeginn=Ende. Der abgeschnittene Abschnitt entspricht der in Zeichen definierten Länge. Wenn der Text zu kurz ist, werden Leerzeichen als Füllzeichen eingefügt.
- FlexText würde normalerweise Trennzeichen (bzw. bei Trennungen mit fester Länge Leerzeichen) zwischen den Elementen einer Trennungsoperation einfügen. Dies ist allerdings bei Trennungen mit der Operation 'getrennt (zeilenbasiert)' nicht der Fall. Die Operation 'getrennt (zeilenbasiert)' kann nicht wieder vollständig rückgängig gemacht werden. Der "getrennte" Text kann an jeder beliebigen Stelle in der ersten Zeile vorkommen und ist im Text inkludiert, daher kann der Text durch einen automatischen Vorgang nicht zuverlässig hinzugefügt werden.
 - mit "getrennt (zeilenbasiert)" wird kein Trennzeichen zur ersten Zeile hinzugefügt, wenn es fehlt.
 - mit "getrennt (beliebig)" wird ein Trennzeichen zwischen zwei Abschnitten hinzugefügt.
- Die Operation "Aufteilen" kann - außer in einfachen Fällen - nicht auf sinnvolle Weise umgekehrt werden. Dabei werden die Verzweigungen nach der ersten Verzweigung durchsucht, die Daten enthält, und diese Daten werden verwendet/eingefügt.

Nur die erste Verbindung einer Aufteilungsoperation wird gemappt. Um Daten in die restlichen "Aufteilen"-Container zu transferieren, müssen Filter für die restlichen Konnektoren definiert werden und das übergeordnete Datenelement der Operation "Aufteilen" muss dupliziert werden, sodass jedes "Aufteilen"-Datenelement ein einziges Datenelement zurückgibt, welches anschließend in ein wiederholtes "Trenn"-Datenelement eingespeist wird, um alle Elemente zusammenzuführen.

- Wenn auf einen Subeintrag eines einzigen Trenn-Containers gemappt wird, werden alle Mapping-Ergebnisse mit Ausnahme des letzten Datenelements verworfen. Nur ein einziges Ergebnis wird beibehalten, selbst wenn mehrere Ergebnisse generiert wurden.

Die folgende Analogie zum XML-Schema Content Model zeigt, wie FlexText sich verhält, wenn es als Zielkomponente verwendet wird:

- eine wiederholte Trennung ist ein wiederholbares Element.
- eine einzige Trennung bildet Teil einer 'sequence' Content Model-Gruppe.
- Bei einer Aufteilung wird eine 'choice' Content Model-Gruppe gebildet, wobei es sich bei jedem Fall um ein mögliches Subelement handelt.
- Mit "Speichern" wird ein Element vom Typ simpleType erstellt.

19.7 FlexText Referenz

Im Referenzabschnitt werden die verschiedenen Funktionen von FlexText beschrieben und Sie erfahren wie Sie sie verwenden, um bestimmte Ergebnisse zu erzielen.

19.7.1 Mehrmals trennen

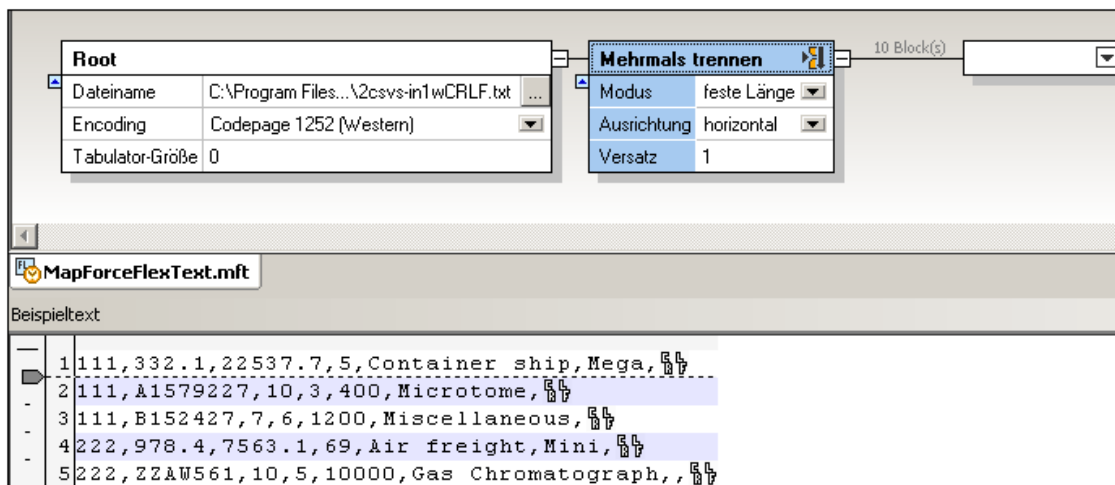


Wenn Sie diese Option am Anfang verwenden, wird ein einzelner Container erstellt. Der Container enthält den Text, der durch die Bedingung definiert wird, die in **Mehrmals trennen** eingestellt ist. Es gibt verschiedene Versionen der Option "Mehrmals trennen"; [Feste Länge](#), getrennt ([beliebig](#)), getrennt [zeilenbasiert](#) und getrennt [Zeile beginnt mit...](#).

Wenn Sie diese Option zum ersten Mal verwenden, werden automatisch Standardparameter verwendet und die erzeugten Fragmente werden im dazugehörigen Container angezeigt. Beachten Sie, dass der "Mehrmals trennen"-Container derzeit aktiv ist, wobei im Beispieltextbereich eine Vorschau aller aktuellen Datensätze/Zeilen angezeigt wird.

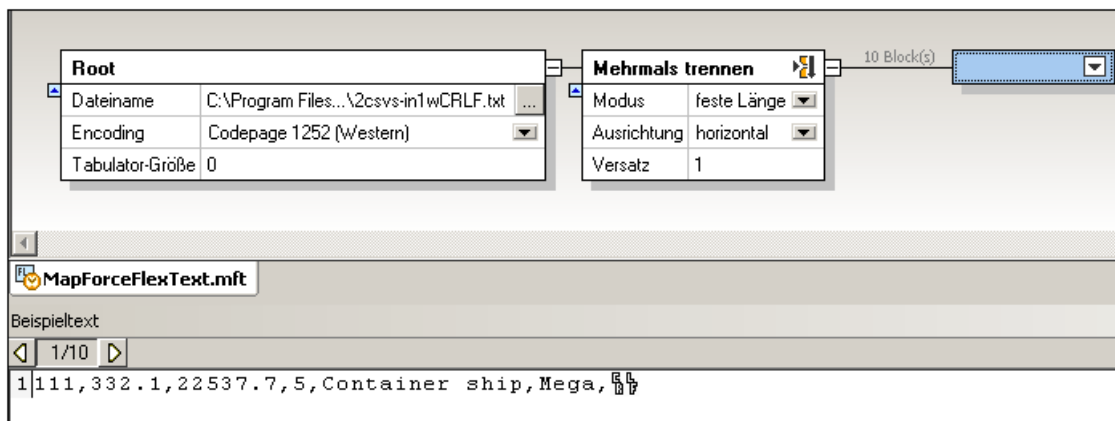
Die **Standardeinstellungen** für den Container sind:

Modus feste Länge
Ausrichtung horizontal
Versatz 1



Standardergebnis:

Jede Textzeile erscheint als Zeile/Datensatz im neuen Container, da als Versatz der Wert 1 angenommen wird. Klicken Sie auf den neuen Container, um eine Vorschau des Inhalts zu sehen. Mit Hilfe der **Bildlaufpfeile** im Beispieltextbereich können Sie durch jeden/jedes der 10 mit diesen Einstellungen erzeugten Blöcke/Fragmente scrollen.



Modus - Feste Länge

Dies ist der Standardwert.

Ausrichtung:

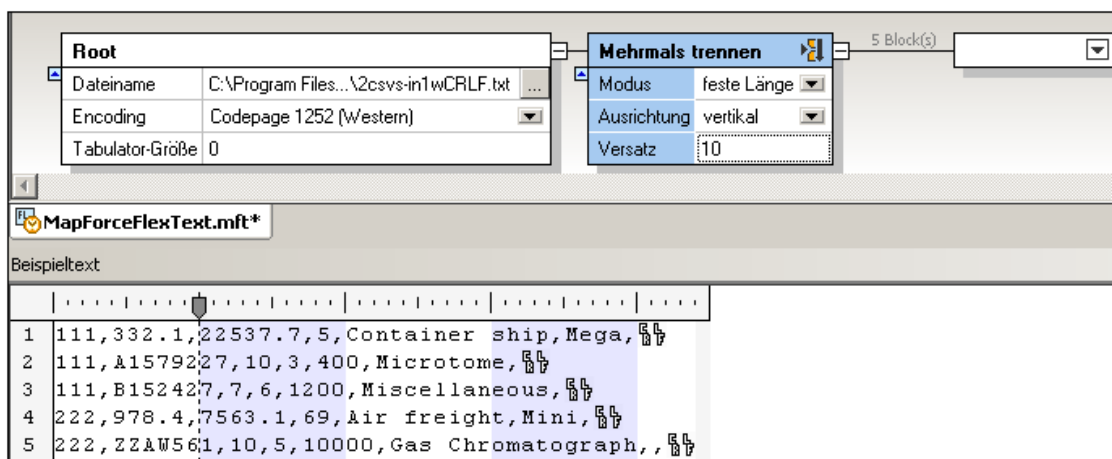
Dient dazu festzulegen, ob das Textfragment nach Zeilen/Datensätzen oder Spalten getrennt werden soll.

Horizontal

Teilt das Fragment in **mehrere** horizontale Abschnitte auf (siehe oben). Geben Sie einen Wert in das Feld "Versatz" ein oder ziehen Sie den Tabulator am vertikalen Lineal.

Vertikal

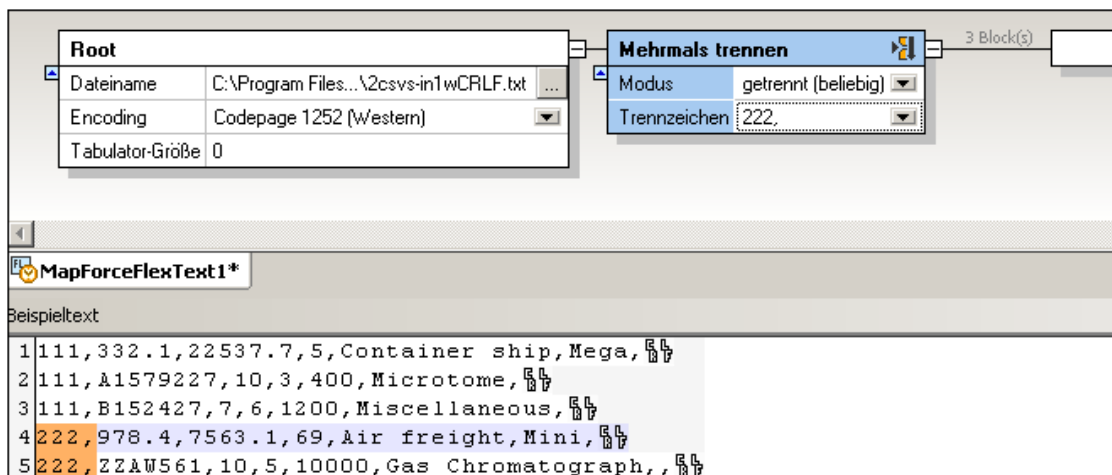
Teilt das Fragment in **mehrere** vertikale Spalten auf. Geben Sie einen Wert in das Feld "Versatz" ein oder ziehen Sie den Tabulator am horizontalen Lineal. Jedes Fragment enthält bis zu Ende der Datei bzw. des Fragments die durch die Versatzbreite definierte Anzahl der Zeichen pro Spalte z.B. 10.



Modus - Getrennt (beliebig)

Die Standardeinstellungen sind:

Trennzeichen (keines)



- Erstellt **mehrere** Fragmente, die durch die Trennzeichen definiert werden, die Sie in das Feld "Trennzeichen" eingeben.
- Die Trennzeichen werden **nicht** in das Fragment eingeschlossen.
- Ein Block/Fragment ist definiert als der Text zwischen dem ersten Zeichen **nach** dem Trennzeichen bis zum letzten Zeichen vor der nächsten Instanz desselben Trennzeichens (mit Ausnahme des ersten und letzten Fragments, siehe unten).

Wenn Sie als Trennzeichen die Zahl "222," verwenden, werden, wie oben gezeigt, 3 separate Fragmente erzeugt:

Fragment 1 enthält alle Zeichen vom **Beginn** des Fragments bis zum Beginn des **ersten Trennzeichens** (222,), also von 111... bis Miscellaneous,.

Beispieltext

1/3

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
3 111,B152427,7,6,1200,Miscellaneous,
```

Wenn das Trennzeichen, wie in diesem Beispiel, nicht die erste Zeichengruppe der ersten Zeile im Fragment ist, enthält das erste Fragment den gesamten Text **bis zur** ersten Instanz des Trennzeichens, also bis 222,.

Wäre 111 das Trennzeichen, dann wäre das erste Fragment ein String mit der Länge Null, da das Trennzeichen am Anfang der ersten Zeile des Quellfragments aufscheint.

Fragment 2 enthält die **erste** Zeile, die das Trennzeichen 222 enthält, ohne das Trennzeichen.

Beispieltext

2/3

```
1 978.4,7563.1,69,Air freight,Mini,
```

Fragment 3 enthält die **nächste** Zeile, die das Trennzeichen 222 enthält, ohne das Trennzeichen selbst, und zwar bis zum Ende der Textdatei/des Textfragments.

Beispieltext

3/3

```
1 ZZAW561,10,5,10000,Gas Chromatograph,,
2
3 General outgassing pollutants
4 1100,897,22.1,716235,LOX
```

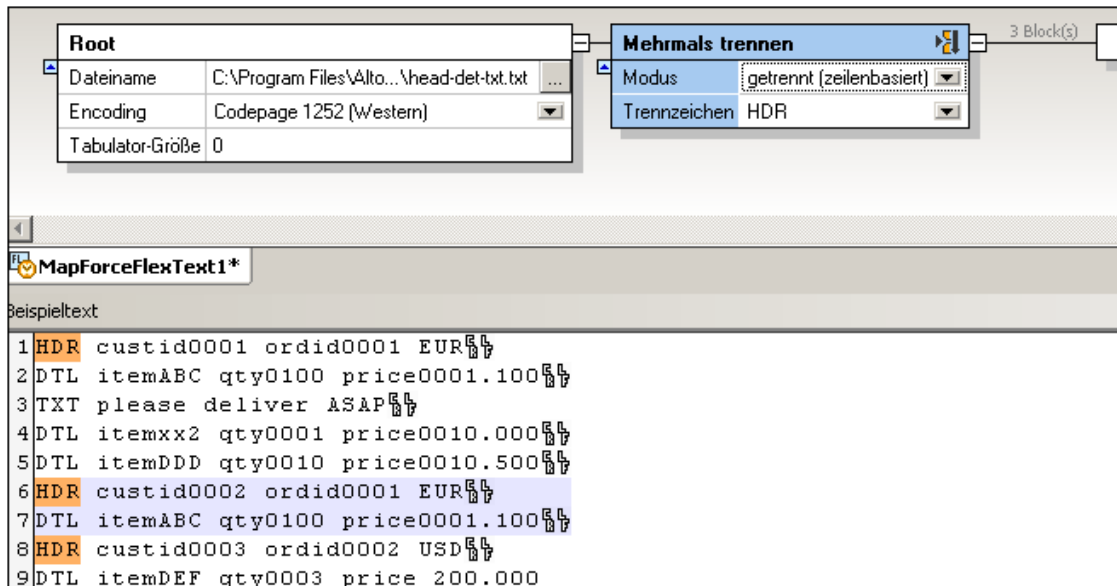
Verwenden Sie diese Option, wenn Sie Fragmente verarbeiten wollen:

- die Trennzeichen enthalten, die Sie daraus entfernen möchten
- Wenn das Fragment vielleicht keine CR/LF-Zeichen enthält und daher Trennzeichen innerhalb der Zeilen enthält.

```
ous,222,978.4,7563.1,69,Air freight,Mini,222,ZZAW561,
```


Modus - Getrennt (zeilenbasiert)

Die Standardeinstellungen sind:
Trennzeichen (keines)



- Mit dieser Option werden mehrere Fragmente erstellt, die durch Trennzeichen definiert werden, die Sie in das Feld "Trennzeichen" eingeben.
- Die Trennzeichen werden in das Fragment **inkludiert**.
- Ein Fragment wird definiert als die **gesamte Zeile**, die das Trennzeichen enthält, bis zur **nächsten Zeile**, die dasselbe Trennzeichen enthält.
- Wenn das Trennzeichen in der ersten **Zeile nicht** vorkommt, so enthält das erste Fragment die Zeile(n) bis zur ersten Zeile, die das Trennzeichen enthält.

Wenn Sie, wie oben gezeigt, das Trennzeichen "**HDR**" verwenden, erhalten Sie 3 separate Fragmente:

Fragment 1 enthält alle Zeichen ab dem Anfang der Datei/des Fragments einschließlich aller Zeilen bis zur nächsten Zeile, die dasselbe Trennzeichen enthält.

```
1 HDR custid0001 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
3 TXT please deliver ASAP
4 DTL itemxx2 qty0001 price0010.000
5 DTL itemDDD qty0010 price0010.500
```

Beachten Sie, dass Sie mit dieser Option jede beliebige Anzahl von Zeilen zwischen zwei Trennzeichen abrufen können, z.B. Dateien mit Überschrift (Header) / Einzelheiten (Detail) / Text, in denen die DTD- oder TXT-Zeilen eventuell optional oder nicht in derselben Reihenfolge sind.

Fragment 2 enthält alle Zeichen/Zeilen ab der zweiten Instanz von HDR bis zur nächsten Instanz von HDR.

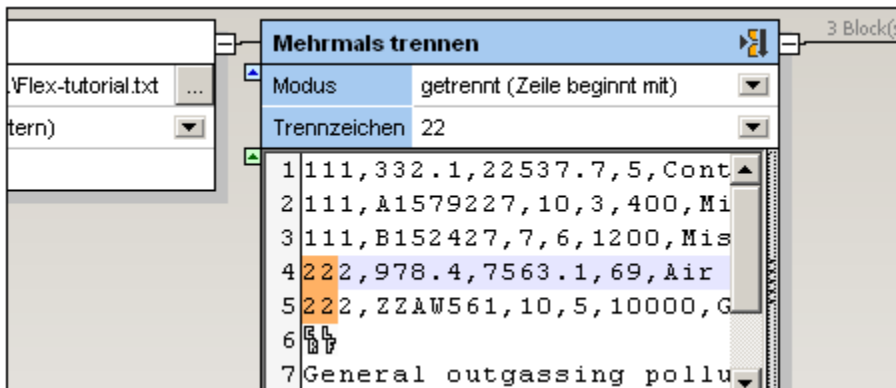
```
1 HDR custid0002 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
```

Fragment 3 enthält alle Zeichen/Zeilen von der dritten Instanz von HDR bis zur nächsten Instanz von HDR.

```
1 HDR custid0003 ordid0002 USD
2 DTL itemDEF qty0003 price 200.000
```

Modus - Getrennt (Beginnt mit...)

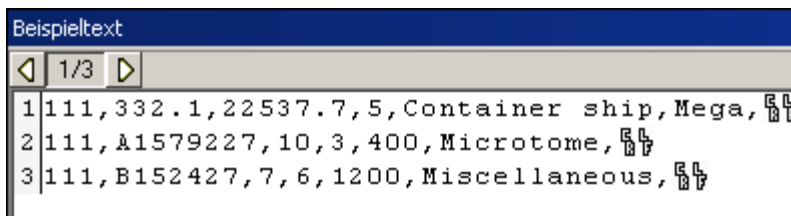
Standardeinstellungen:
Trennzeichen (keines)



- Erstellt mehrere Fragmente, die Sie durch Trennzeichen definieren, die Sie in das Feld "Trennzeichen" eingeben.
- Die Trennzeichen werden in das Fragment **inkludiert**.
- Ein Fragment wird definiert als die **gesamte Zeile, beginnend** mit dem Trennzeichen bis zur **nächsten Zeile**, die das gleiche Trennzeichen am **Beginn** der Zeile enthält.
- Wenn in der ersten **Zeile kein Trennzeichen** vorkommt, so enthält das erste Fragment die Zeile(n) bis zur ersten Zeile, die das Trennzeichen enthält.

Bei Verwendung des Trennzeichens "22" (wie oben gezeigt) erhalten Sie 3 separate Fragmente:

Fragment 1, enthält alle Zeichen vom Beginn der Zeile/des Fragments bis zu der Zeile, die das Trennzeichen 22 enthält.



Fragment 2, enthält alle Zeichen/Zeilen ab der Stelle, an der das Trennzeichen 22 das zweite Mal vorkommt, bis zu der Stelle, an der 22 das nächste Mal vorkommt. Dies ist in diesem Fall nur eine Zeile.

Beispieltext

2/3

```
1 222,978.4,7563.1,69,Air freight,Mini, 
```

Fragment 3, enthält alle Zeichen/Zeilen ab der Stelle, an der 22 das dritte Mal vorkommt bis zum Ende der Datei/des Fragments.

Beispieltext

3/3

```
1 222,ZZAW561,10,5,10000,Gas Chromatograph,, 
2 
3 General outgassing pollutants 
4 1100,897,22.1,716235,LOX 
5 1110,9832,22991.30,002,NOX 
6 1120,1213,33.01,008,SOX
```

Vergleichen Sie das obige Beispiel mit dem Ergebnis, das wir erhalten würden, wenn wir die Option **getrennt (zeilenbasiert)** und das Trennzeichen **22** verwenden würden.

Mehrmals trennen

Modus: getrennt (zeilenbasiert)

Trennzeichen: 22

Beispieltext

```
1 111,332.1,22537.7,5,Container ship,Mega, 
2 111,A1579227,10,3,400,Microtome, 
3 111,B152427,7,6,1200,Miscellaneous, 
4 222,978.4,7563.1,69,Air freight,Mini, 
5 222,ZZAW561,10,5,10000,Gas Chromatograph,, 
6 
7 General outgassing pollutants 
8 1100,897,22.1,716235,LOX 
9 1110,9832,22991.30,002,NOX
```

In diesem Fall gäbe es sechs Fragmente bestehend aus Zeilen, die die Zahl 22 irgendwo in der Zeile enthalten.

19.7.2 Einmal trennen

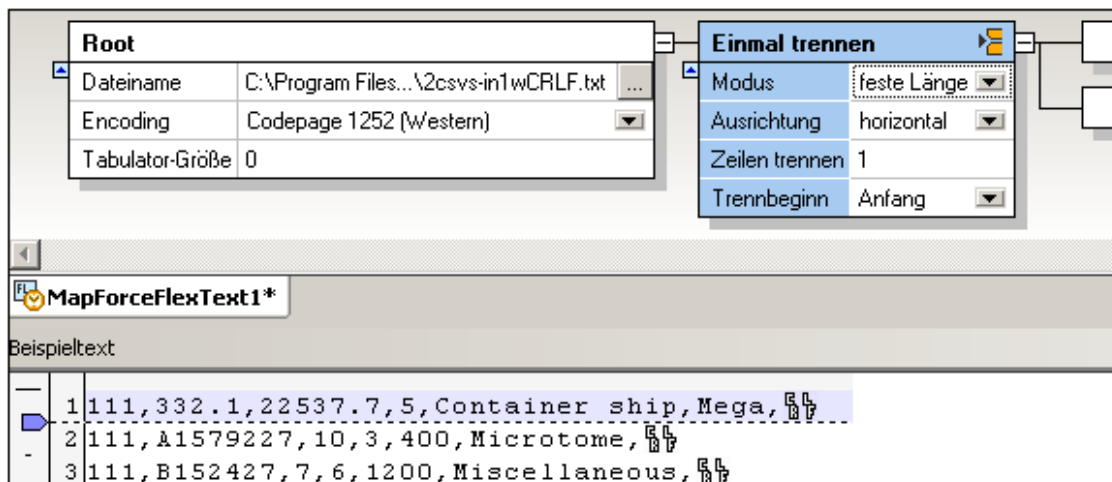


Mit dieser Option werden zwei vertikal übereinander ausgerichtete Container angelegt. Der obere Container enthält den Text, der durch die Bedingung definiert wird, die im **Einmal trennen**-Container eingestellt wurde. Der untere Container enthält den Rest der Textdatei/des Fragments. Es gibt mehrere Versionen der Option "Einmal trennen"; [Feste Länge](#), getrennt [\(beliebig\)](#) und getrennt [zeilenbasiert](#).

Wenn Sie diese Option zum ersten Mal auswählen, werden automatisch Standardparameter verwendet und die erzeugten Fragmente werden in beiden Containern angezeigt. Beachten Sie, dass der **Einmal trennen**-Container derzeit aktiv ist und im Beispieltextbereich eine Vorschau aller aktuellen Datensätze/Zeilen angezeigt wird.

Die **Standardeinstellungen** für den Container sind:

Modus	feste Länge
Ausrichtung	horizontal
Zeilen trennen	1
Trennbeginn	Anfang



Standardergebnis:

Die erste Zeile des Texts erscheint im oberen Container. Der untere Container enthält den Rest der Textdatei/des Fragments.

Modus - Feste Länge

Dies ist der Standardwert.

Ausrichtung:

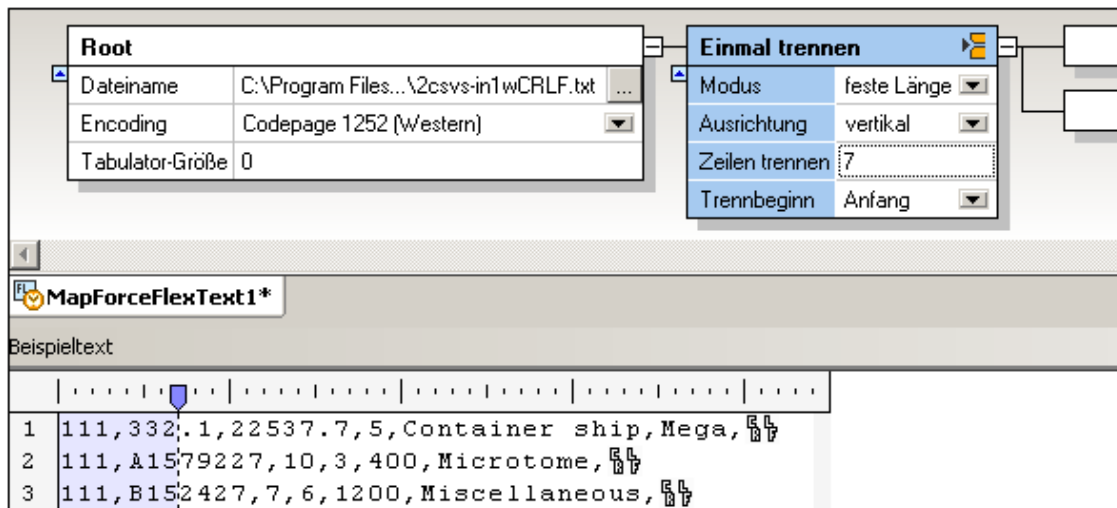
Damit können Sie definieren, wie das Textfragment getrennt werden soll: nach Zeilen/Datensätzen oder Spalten.

Horizontal

Teilt das Fragment in **zwei** horizontale Abschnitte auf. Geben Sie einen Wert in das Feld "Zeilen trennen" ein oder ziehen Sie den Tabulator am vertikalen Lineal.

Vertikal

Teilt das Fragment in **zwei** vertikale Spalten auf. Geben Sie einen Wert in das Feld "Zeilen trennen" ein oder ziehen Sie den Tabulator am horizontalen Lineal.

**Zeilen trennen:**

Die Anzahl der Zeilen, nach der das Fragment in zwei Fragmente aufgeteilt werden soll.

Trennbeginn:

Anfang: trennt das Fragment nach der definierten Anzahl der Zeilen vom Anfang des Fragments an.

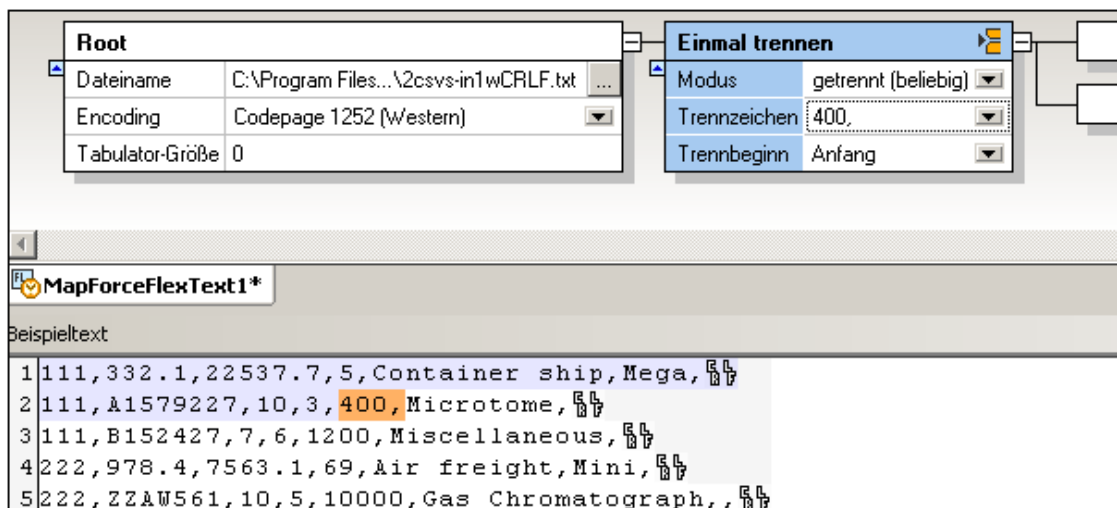
Ende: trennt das Fragment nach der definierten Anzahl der Zeilen vom Ende des Fragments an (verwenden Sie diese Option, wenn die Anzahl der Zeilen/Datensätze nicht bekannt ist).

Modus - Getrennt (beliebig)

Die Standardeinstellungen sind:

Trennzeichen (keines)

Trennbeginn Anfang



- Mit dieser Option werden **zwei** Fragmente erzeugt, die durch die Trennzeichen definiert werden, die Sie in das Feld "Trennzeichen" eingegeben haben.
- Die Trennzeichen werden **nicht** in das Fragment inkludiert.
- Das erste Fragment ist definiert als der Text zwischen dem ersten Zeichen der Datei/des Fragments bis zum letzten Zeichen vor dem Trennzeichen.
- Das zweite Fragment ist definiert als das erste Zeichen nach dem "Trennzeichen" bis

- zum letzten Zeichen in der Datei/im Fragment.
- Wenn das Trennzeichen an der ersten/letzten **Position** der Datei/des Fragments vorkommt, so bleibt der obere Container leer.

Verwenden Sie diese Methode, wenn Sie **einen Abschnitt** einer Datei bzw. eines Fragments abtrennen möchten, wobei sich das Trennzeichen an einer beliebigen Stelle in der Datei/im Fragment befindet. Diese Option eignet sich im Allgemeinen für Dateien, die keine CR- oder LF-Zeichen enthalten und die Sie an einer bestimmten Stelle innerhalb einer Zeile teilen möchten.

Das **obere** Fragment enthält den Text bis zum Trennzeichen (d.h. 400,)

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,
```

Das **untere** Fragment enthält die verbleibenden Zeichen nach dem Trennzeichen.

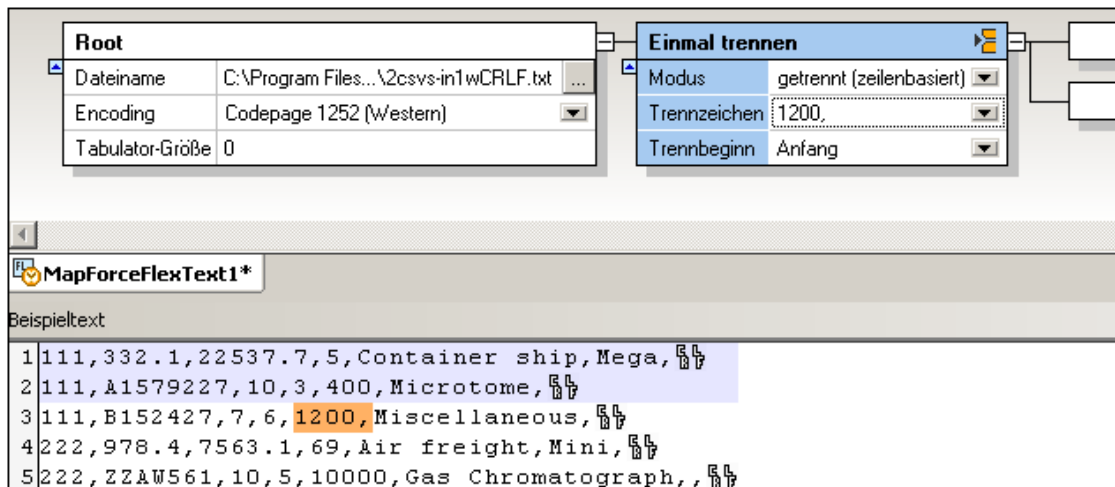
```
1 Microtome,
2 111,B152427,7,6,1200,Miscellaneous,
3 222,978.4,7563.1,69,Air freight,Mini,
4 222,ZZAW561,10,5,10000,Gas Chromatograph,,
```

Modus - Getrennt (zeilenbasiert)

Die Standardeinstellungen sind:

Trennzeichen (keines)

Trennbeginn Anfang



- Mit dieser Option werden **zwei** Fragmente erzeugt, die durch die Trennzeichen, die Sie in das Feld "Trennzeichen" eingeben, definiert werden.
- Die Trennzeichen werden im Fragment **inkludiert**.
- Das erste Fragment ist definiert als der gesamte Text bis zur Zeile, die das Trennzeichen enthält.
- Das zweite Fragment ist definiert als der Text und die Zeile einschließlich des Trennzeichens bis zum Ende der Datei/des Fragments.
- Wenn das Trennzeichen in der ersten/letzten **Zeile** der Datei/des Fragments vorkommt, so bleibt der obere Container leer.

Verwenden Sie diese Methode, um eine Datei oder ein Fragment entzweizuteilen, in der/dem sich das Trennzeichen an einer beliebigen Stelle in einer der Zeilen befindet. Die Zeile, die das

Trennzeichen enthält, wird nicht aufgeteilt, sondern bleibt ganz. Diese Option eignet sich im Allgemeinen für Dateien, die Datensatztrennzeichen (CR/LF) enthalten und die Sie in zwei separate Fragmente aufteilen möchten.

Das **obere** Fragment enthält den Text **bis zur Zeile**, die das Trennzeichen enthält.

1	111,332.1,22537.7,5,Container ship,Mega,
2	111,A1579227,10,3,400,Microtome,

Das **untere** Fragment enthält die gesamte Zeile, die das Trennzeichen enthält (1200,) sowie alle restlichen Zeilen bis zum Ende der Datei/des Fragments.

1	111,B152427,7,6,1200,Miscellaneous,
2	222,978.4,7563.1,69,Air freight,Mini,
3	222,ZZAW561,10,5,10000,Gas Chromatograph,,

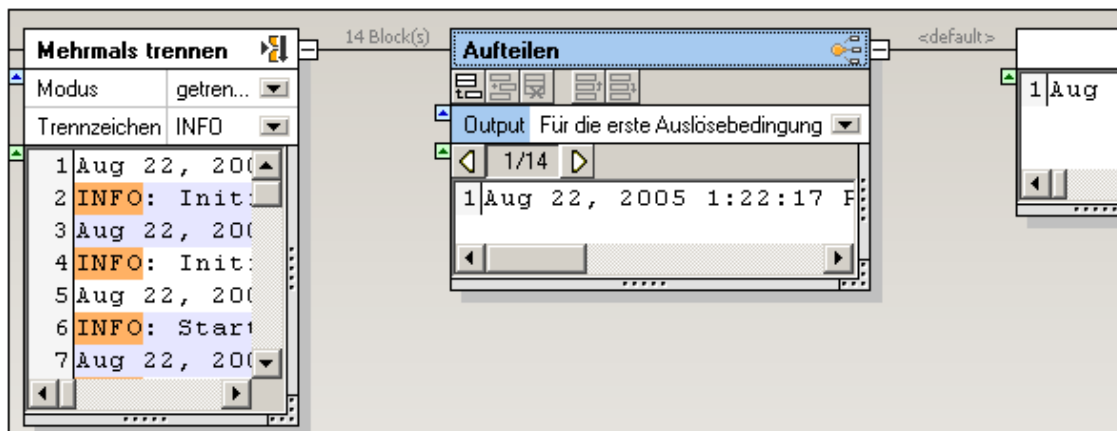
19.7.3 Aufteilen



Mit Hilfe der Option "Aufteilen" können Sie für ein einzelnes Textfragment mehrere Schlüsselwörter oder Bedingungen definieren. Für jedes Schlüsselwort, das Sie definieren, gibt es einen eigenen Container, in dem die Daten nur dann abgelegt werden, wenn eine bestimmte Bedingung zutrifft, also "true" ist. Wenn keiner der Bedingungen zutrifft, so wird das jeweilige Fragment in einem "Standardcontainer" abgelegt.

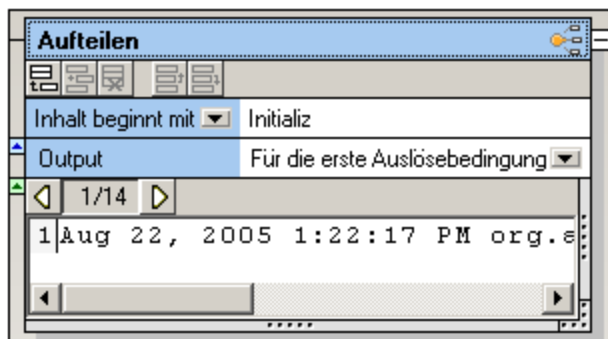
Die **Standardeinstellungen** für den Container sind:
Output: für die erste Auslösebedingung.

Im Beispiel unten wird eine Tomcat-Log-Datei verarbeitet, in der die einzelnen Prozesse herausgetrennt werden sollen, damit sie gemappt werden können. Wenn Sie einen "Aufteilen"-Container zum ersten Mal definieren, erscheint rechts neben dem "Aufteilen"-Container nur der Standardcontainer. Alle Daten werden automatisch an diesen Container übergeben.



Im "Mehrmals trennen"-Container werden unter Verwendung der Option "getrennt (zeilenbasiert)" alle INFO-Abschnitte aus der Log-Datei herausgetrennt und an den "Aufteilen"-Container übergeben.

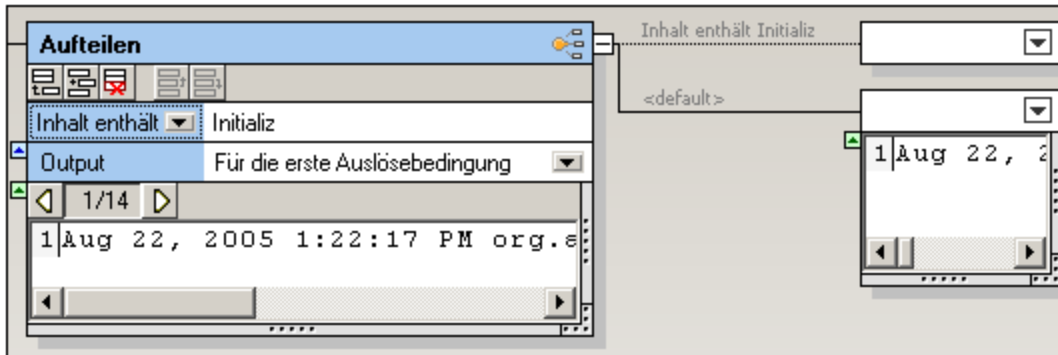
1. Klicken Sie auf das Symbol "Anhängen" , um eine neue Bedingung zum "Aufteilen"-Container hinzuzufügen.
2. Doppelklicken Sie auf das Feld rechts neben "**Inhalt beginnt mit**" geben Sie "Initializ" ein und drücken Sie die Eingabetaste.



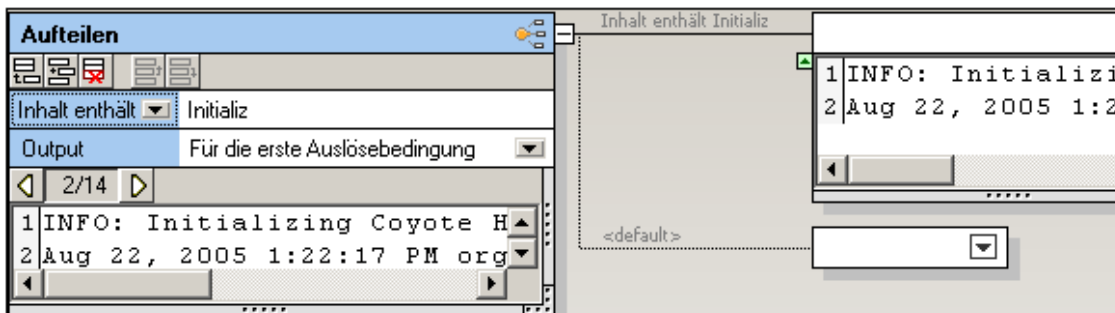
Es wird ein neuer Container hinzugefügt. Wenn die Bedingung zutrifft, also "true" ist, werden die Daten an diesen Container übergeben. Wenn nicht, werden sie an den

Standard-Container übergeben.

3. Klicken Sie auf das Listenfeld "Inhalt beginnt mit" und ändern Sie den Eintrag in "Inhalt enthält".
Die erste Bedingung wurde nun definiert und Sie sehen das Resultat unten.
Das erste Fragment enthält "Initializ" nicht, daher wird der Inhalt an den **Standard-**Container übergeben.



4. Klicken Sie auf das Symbol "Nächsten Block anzeigen" (Next Block Show) , um das nächste Textfragment zu sehen.

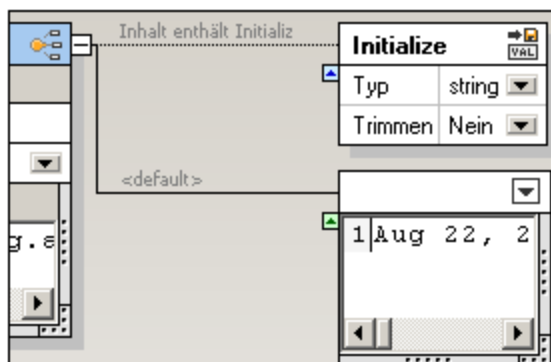


Das Fragment "Initializing..." erscheint nun im dazugehörigen Container und der Standardcontainer ist leer. Wenn Sie die Fragmente der Reihe nach anzeigen, sehen Sie, was die einzelnen Container enthalten werden.

5. Klicken Sie auf das Containersymbol und wählen Sie den Befehl **Als Wert speichern**.

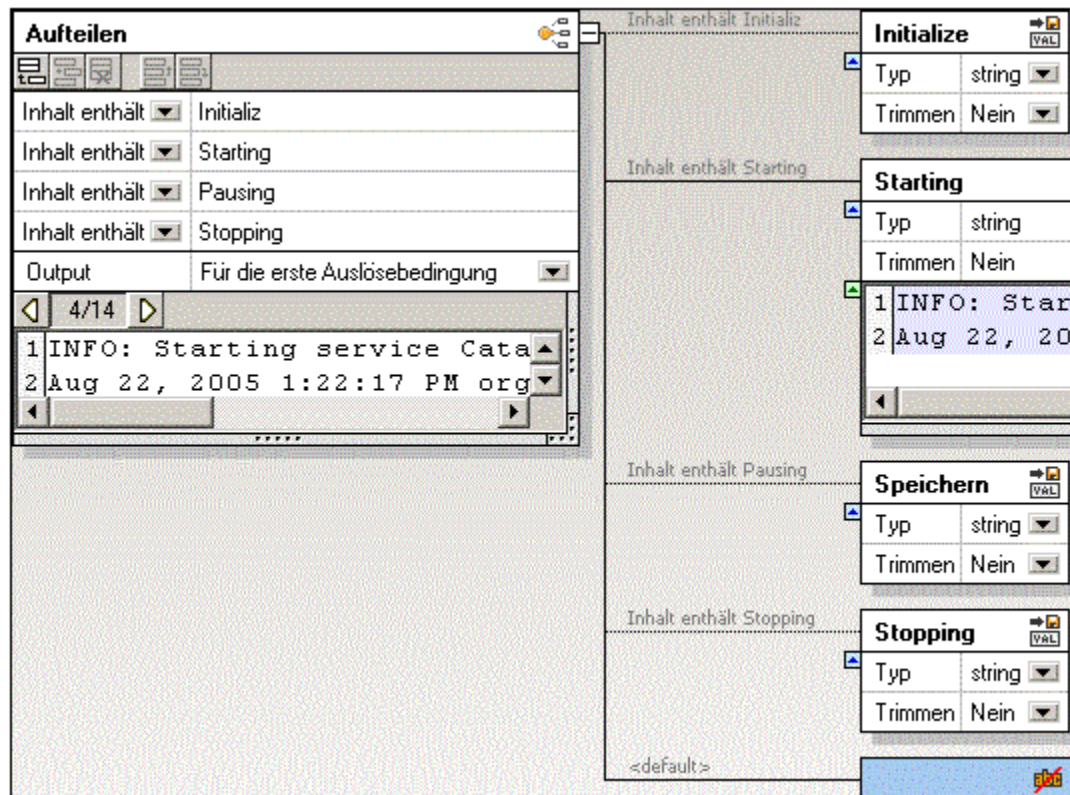


6. Doppelklicken Sie auf die Titelleiste "Speichern" und ändern Sie den Text z.B. Initialize.



7. Klicken Sie auf das Symbol "Anhängen" (Append) , um eine neue Bedingung zum "Aufteilen"-Container hinzuzufügen.

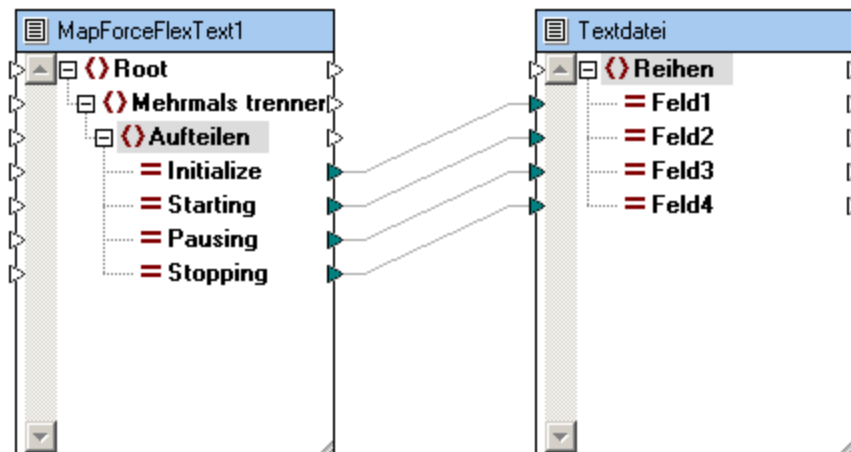
8. Doppelklicken Sie in das Feld neben **"Inhalt beginnt mit"**, geben Sie den Text "Starting" ein und drücken Sie die Eingabetaste.
 Sie können beliebig viele Bedingungen hinzufügen, z.B. Pausing und Stopping.
 Geben Sie jedem der dazugehörigen Container einen Namen, um sie in MapForce leichter unterscheiden zu können.



Im obigen Screenshot sehen Sie alle vier Bedingungen und den Inhalt des "Starting"-Containers bei Block/Fragment Nr. 6. Die entsprechenden Container wurden alle umbenannt, um sie in MapForce besser unterscheiden zu können.

Beachten Sie, dass die Bedingungen in der Liste der Bedingungen mit Hilfe der entsprechenden Schaltflächen (Nach oben/Nach unten  bzw. ) nach oben oder nach unten verschoben werden können.

9. Speichern Sie die Vorlage und fügen Sie sie in MapForce ein.



Bitte beachten Sie:

Wenn ein Textfragment im aktuellen Fragment eine Bedingung erfüllt, so werden **alle Daten** dieses Fragments an den dazugehörigen Container übergeben. Die Daten werden nicht getrennt, sondern nur in den entsprechenden Container bzw. - wenn keine der definierten Bedingungen zutrifft - in den Standardcontainer geroutet.

Die mit "Aufteilen" erzeugten Container können für die weitere Verarbeitung verwendet werden. Sie können einen solchen Container in "Einmal trennen", "Mehrmals trennen" oder eine beliebige andere Option ändern.

Inhalt beginnt mit:

Die Daten werden nur dann an den dazugehörigen Container übergeben, wenn die in der Bedingung definierte Zeichenfolge am Anfang des Textfragments vorkommt.

Inhalt enthält:

Die Daten werden an den dazugehörigen Container übergeben, wenn die in der Bedingung definierte Zeichenfolge an einer beliebigen Stelle im Textfragment vorkommt.

Für die erste Auslösebedingung:

Die Daten werden übergeben, wenn **eine** der **Bedingungen** in der Bedingungsliste **true** ist. Alle anderen Bedingungen, die "true" sind, werden ignoriert und es werden keine Daten an einen der dazugehörigen Container übergeben.

Für alle Auslösebedingungen:

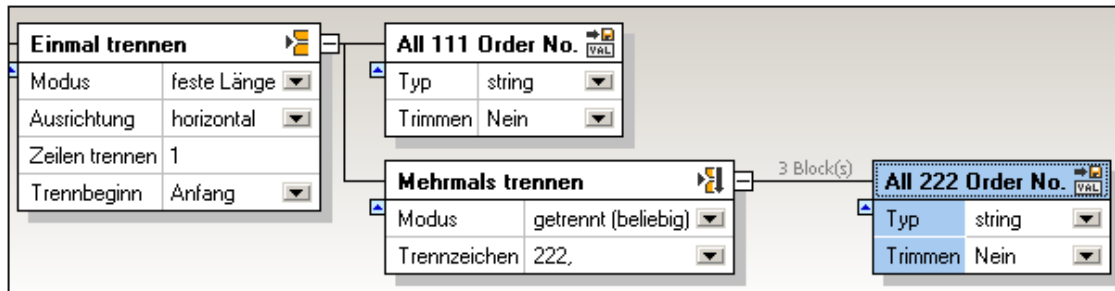
Es werden für **jede** Bedingung in der Bedingungsliste, die "true" ist, Daten übergeben. Auf diese Art wird es möglich, dass mehrere Instanzen derselben Daten/desselben Fragments in mehreren verknüpften Containern gleichzeitig vorkommen. Dies kann vorkommen, wenn ein Textfragment Text enthält, der gleichzeitig zwei Bedingungen erfüllt z.B. "initializing starting sequence" im obigen Beispiel.

19.7.4 Node

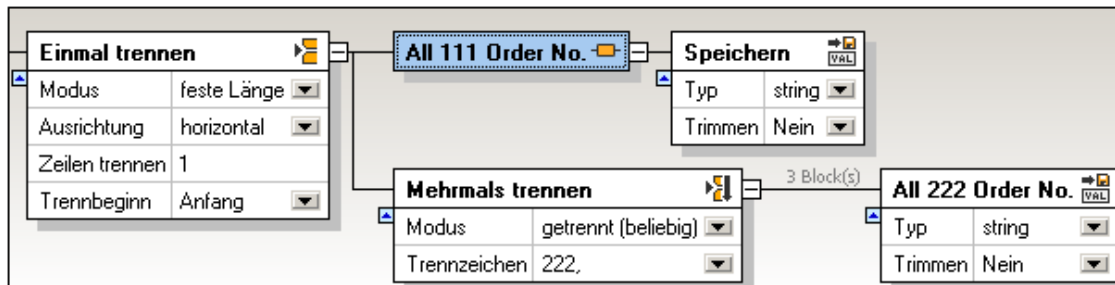


Damit können Sie eine neue hierarchische Ebene zur FlexText- und MapForce-Baumstruktur hinzufügen. Die Daten, die der folgende Node/Container enthält, werden so übergeben, wie sie sind.

Im unten gezeigten Screenshot ist der Container "All 111 Order No." der letzte Container im oberen Zweig.



Klicken Sie auf das Symbol dieses Containers und wählen Sie die Node-Option aus dem Popup-Menü aus.

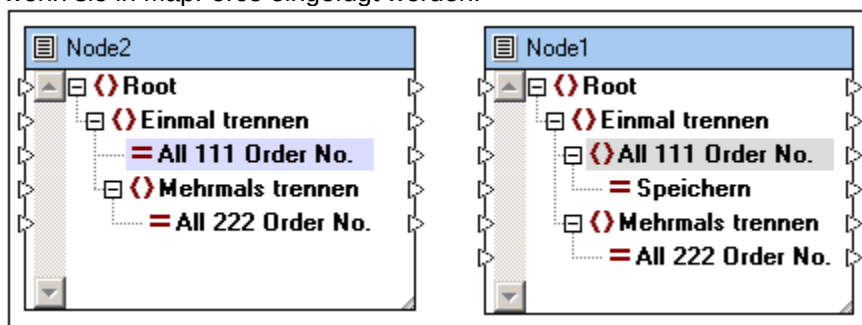


Es wurde nun ein neuer Container rechts vom aktuellen hinzugefügt.

Bitte beachten Sie:

Der automatisch hinzugefügt Container wurde anschließend manuell als **"Als Wert speichern"** definiert.

Im unten gezeigten Screenshot sehen Sie beide Vorlagenstrukturen so, wie sie aussehen, wenn sie in MapForce eingefügt werden.



In der linken Komponente sehen Sie die anfängliche Struktur, bevor der neue Node hinzugefügt wurde.

In der rechten Komponente sehen Sie, wie sich die Komponentenstruktur geändert hat. "All 111..." ist nun ein Parent-Element. Unterhalb davon wurde ein neues Child-Element "Speichern"

hinzugefügt.

19.7.5 Ignorieren



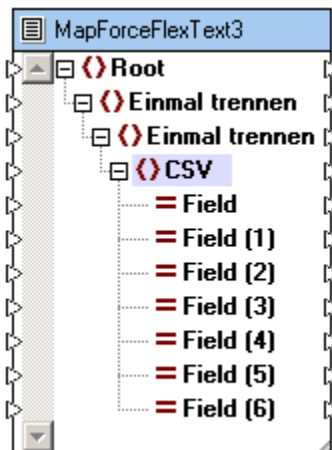
Mit dieser Option können Sie die Ausgabe eines bestimmten Textfragments unterdrücken. D.h. dieser Container und alle Daten, die er enthält, stehen in der FlexText-Komponente in MapForce nicht für das Mapping zur Verfügung.

Beispieltext

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
3 111,B152427,7,6,1200,Miscellaneous,
```

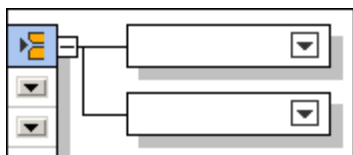
Im obigen Beispiel wurde der aktive Container auf "Ignorieren" eingestellt. Daher steht der darin enthaltene Beispieltext in MapForce nicht für das Mapping zur Verfügung.

Wenn Sie die Textvorlage in MapForce einfügen, hat sie die unten gezeigte Struktur. Zwischen den beiden "Einmal trennen"-Elementen gibt es kein Element, das gemappt werden kann.



Bitte beachten Sie:

Es gibt auch Standard "Ignorieren"-Container. Dies sind die neuen Container, die automatisch angehängt werden, wenn Sie "Einmal trennen" oder "Mehrals trennen" usw. auswählen.



Der Inhalt dieser Container kann anfangs in MapForce nicht gemappt werden, wenn die Vorlage eingefügt wird. Sie müssen zuerst in FlexText eine der Container-Optionen auswählen: Als Wert speichern, Als CSV speichern usw., um die Container mappen zu können.

19.7.6 Als CSV speichern (getrennt)



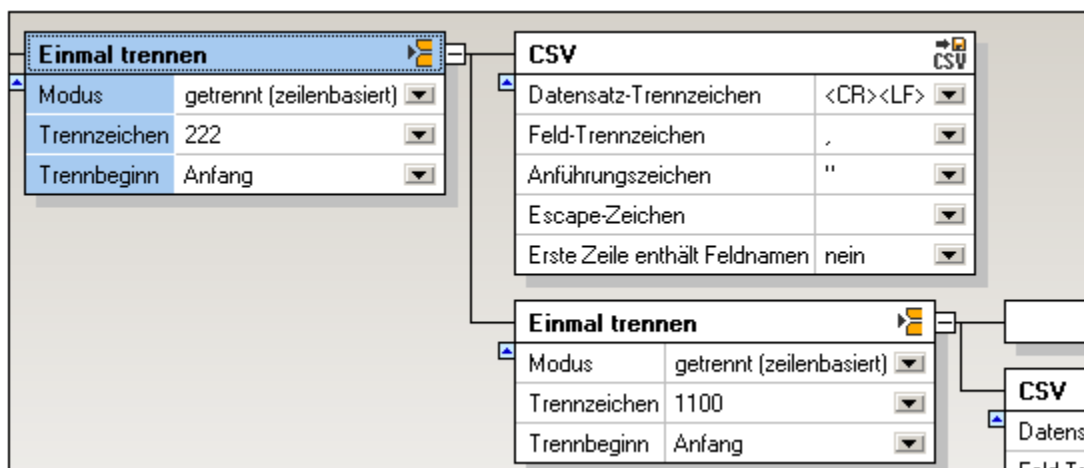
Mit dieser Option können Sie Textfragmente als CSV-Text speichern und einzelne Spalten in MapForce mappen. Sie können in FlexText beliebig viele CSV-Container/Komponenten erstellen, wobei jeder der CSV-Container andere Trennzeichen enthalten kann.

Der Beispieltextbereich bietet eine Übersicht über das aktuelle CSV-Fragment und gestattet Ihnen, einzelne Feldnamen und Feldtypen zu definieren. Jede Spalte wird in der FlexText-Komponente in MapForce als mapbares Element dargestellt.

Die **Standardeinstellungen** des Containers sind:

Datensatz-Trennzeichen	CR LF
Feld-Trennzeichen	,
Anführungszeichen	"
Escape-Zeichen	(keines)
Erste Zeile enthält Feldnamen	nein

Im folgenden Beispiel sehen Sie, wie Daten in einer kleinen Textdatei in zwei CSV-Dateien aufgeteilt und in MapForce auf separate XML-Dateien gemappt werden.



Der oben gezeigte **Einmal trennen**-Container wird verwendet, um zwei Container zu erzeugen. Dies wird mit der Option **getrennt (zeilenbasiert)** und unter Verwendung des Trennzeichens 222 erzielt. Alle Datensätze bis zur ersten Instanz von 222 werden an den CSV-Container übergeben. Der erste Container, der alle Datensätze mit der Zahl 111 enthält, wird anschließend als ein CSV-Container definiert. Im Beispieltextbereich sehen Sie den Inhalt des derzeit aktiven "Einmal trennen"-Containers.


```

Beispieltext
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
3 111,B152427,7,6,1200,Miscellaneous,
4 222,978.4,7563.1,69,Air freight,Mini,
5 222,ZZAW561,10,5,10000,Gas Chromatograph,,
6
7 General outgassing pollutants
8 1100,897,22.1,716235,LOX
9 1110,9832,22991.30,002,NOX
10 1120,1213,33.01,008,SOX

```

Die Standard-CSV-Einstellungen wurden nicht geändert. Wenn Sie auf den CSV-Container klicken, sehen Sie seinen Inhalt in Tabellenform.

Einmal trennen

Modus	getrennt (zeilenbasiert)
Trennzeichen	222
Trennbeginn	Anfang

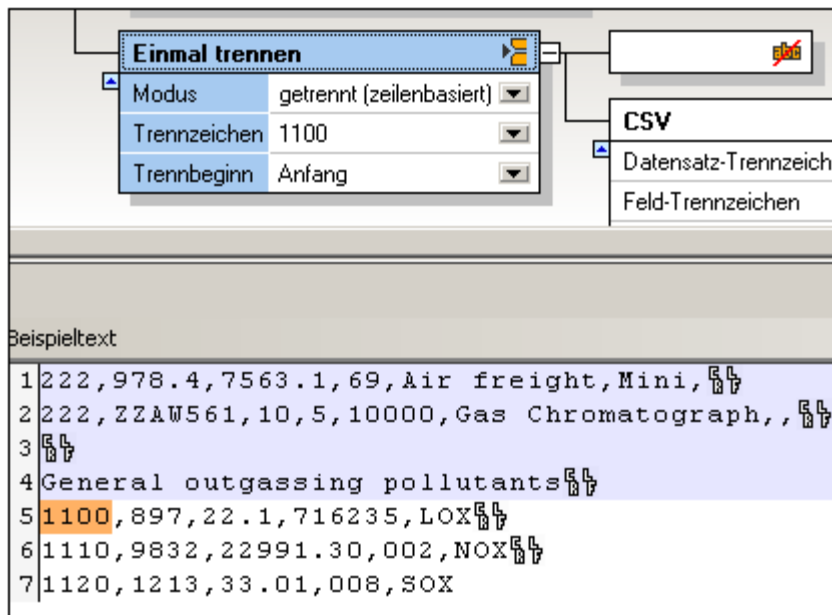
CSV

Datensatz-Trennzeichen	<CR><LF>
Feld-Trennzeichen	,
Anführungszeichen	"
Escape-Zeichen	
Erste Zeile enthält Feldnamen	nein

Beispieltext

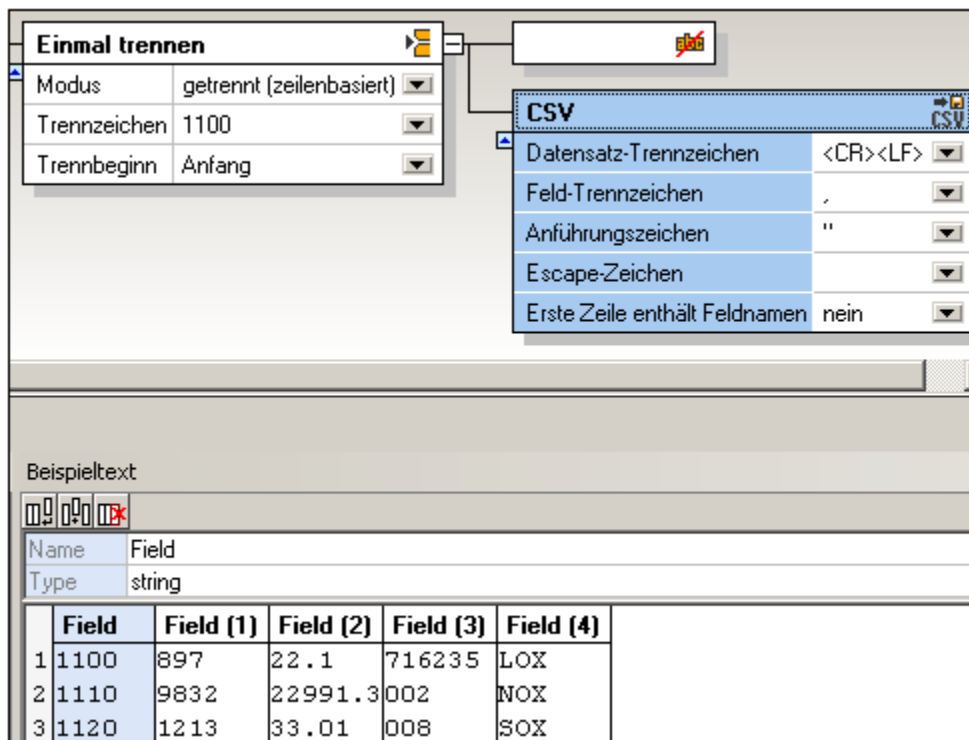
Name	Field					
Type	string					
Field	Field (1)	Field (2)	Field (3)	Field (4)	Field (5)	Field (6)
1 111	332.1	22537.7	5	Contain	Mega	
2 111	A157922	10	3	400	Microto	
3 111	B152427	7	6	1200	Miscell	

Der zweite Container enthält die restlichen Daten und wird als ein weiterer **Einmal trennen**-Container definiert. Dadurch werden zwei weitere Container erzeugt, von denen einer der zweite CSV-Container sein wird. Wenn Sie auf den "Einmal trennen"-Container klicken, sehen Sie den aktuellen Inhalt.

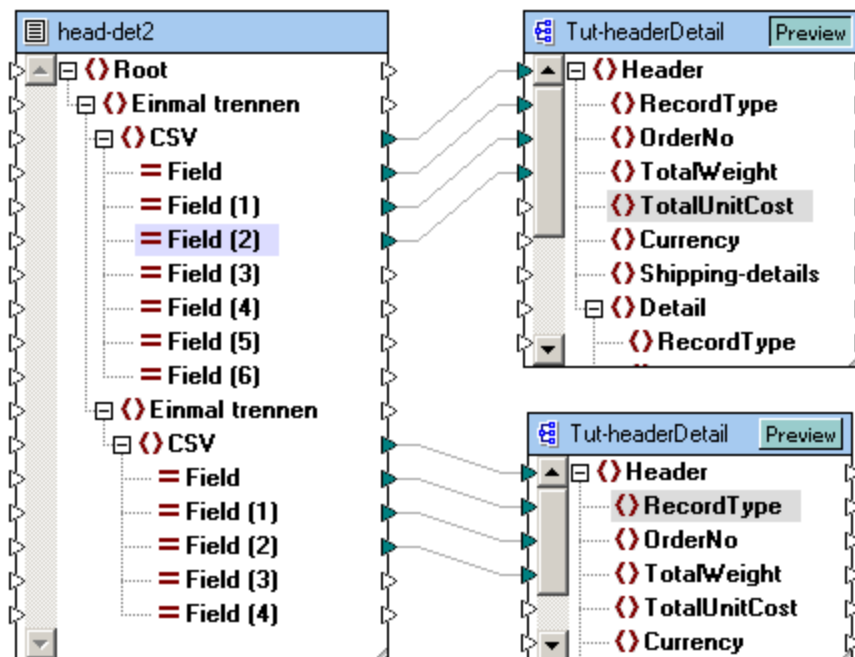


Mit Hilfe der Funktion "getrennt (zeilenbasiert)" und dem Trennzeichen "1100" werden die restlichen Daten in zwei Abschnitte aufgeteilt.

- Alle Datensätze bis zur ersten Instanz von 1100 werden an den ersten Container übergeben, der mit "Ignorieren" als nicht mapbar definiert wird.
- Anschließend wird der zweite Container als CSV definiert. Die Standardeinstellungen wurde nicht geändert. Wenn Sie auf den CSV-Container klicken, wird dessen Inhalt in Tabellenform angezeigt.



Wenn Sie die FlexText-Vorlage in MapForce einfügen, können Sie die Daten auf jede der unterstützten Zieldateien mappen. In diesem Beispiel wird jedes der CSV-Elemente auf zwei separate XML-Dateien gemappt.



Beachten Sie, dass nicht alle der Elemente in den CSV-Abschnitten auf die Zieldateien gemappt werden.

Die erste XML-Datei enthält alle 111 Datensatztypen.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3  <Header>
4  <RecordType>111</RecordType>
5  <OrderNo>332</OrderNo>
6  <TotalWeight>22537.7</TotalWeight>
7  </Header>
8  <Header>
9  <RecordType>111</RecordType>
10 <OrderNo>0</OrderNo>
11 <TotalWeight>10</TotalWeight>
12 </Header>
13 <Header>
14 <RecordType>111</RecordType>
15 <OrderNo>0</OrderNo>
16 <TotalWeight>7</TotalWeight>
17 </Header>
18 </Order>
19

```

Die zweite XML-Datei enthält alle Datensätze beginnend mit 1100.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <Header>
4      <RecordType>1100</RecordType>
5      <OrderNo>897</OrderNo>
6      <TotalWeight>22.1</TotalWeight>
7      <TotalUnitCost>716235</TotalUnitCost>
8    </Header>
9    <Header>
10     <RecordType>1110</RecordType>
11     <OrderNo>9832</OrderNo>
12     <TotalWeight>22991.3</TotalWeight>
13     <TotalUnitCost>2</TotalUnitCost>
14   </Header>
15   <Header>
16     <RecordType>1120</RecordType>
17     <OrderNo>1213</OrderNo>
18     <TotalWeight>33.01</TotalWeight>
19     <TotalUnitCost>8</TotalUnitCost>
20   </Header>
21 </Order>
22

```

Konfigurieren der CSV Container/Daten:

Einmal trennen

Modus	getrennt (zeilenbasiert)
Trennzeichen	1100
Trennbeginn	Anfang

CSV

Datensatz-Trennzeichen	<CR><LF>
Feld-Trennzeichen	,
Anführungszeichen	"
Escape-Zeichen	
Erste Zeile enthält Feldnamen	nein

Beispieltext

Name	Field
Type	string
Field	
Field (1)	
Field (2)	
Field (3)	
Field (4)	

	Field	Field (1)	Field (2)	Field (3)	Field (4)
1	1100	897	22.1	716235	LOX
2	1110	9832	22991.3	002	NOX
3	1120	1213	33.01	008	SOX

Wenn Sie im Beispieltextbereich auf ein Feld klicken, wird dieses markiert, sodass Sie es näher konfigurieren können.

- Klicken Sie in das Feld **Name**, um den hier erscheinenden Standardtext zu editieren.
- Klicken Sie in das Feld **Type**, um die Felddatentypen zu definieren: string, boolean, decimal usw.
- Klicken Sie auf das Symbol "Anhängen" , um ein neues Feld anzuhängen.

- Klicken Sie auf das Symbol "Einfügen" , um vor dem derzeit aktiven Feld ein Feld einzufügen.
- Klicken Sie auf das Symbol "Löschen" , um das derzeit aktive Feld zu löschen.

Bitte beachten Sie:

Sie können die Ränder des Felds mit der Maus ziehen, um die Daten zu sehen.

Leere Felder als nicht vorhanden behandeln

Damit können Sie definieren, dass für leere Felder in der Quelldatei keine entsprechenden leeren Datenelemente (Element oder Attribut) in der Zieldatei erzeugt werden.

Beachten Sie, dass dennoch Trennzeichen für die leeren Felder in der Quelldatei vorhanden sein müssen, z.B. "General outgassing pollutants,,,".

19.7.7 Als FLF speichern (feste Länge)

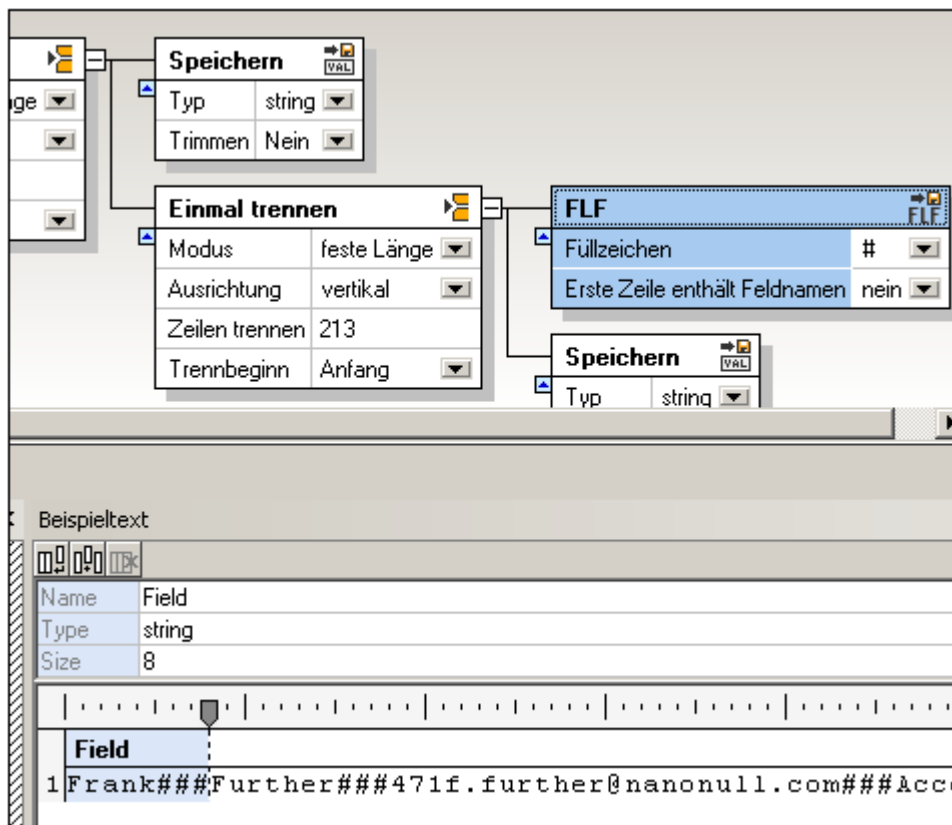


Mit dieser Option können Sie Textfragmente als Text mit fester Länge speichern und einzelne Spalten auf MapForce mappen. Sie können in FlexText beliebig viele FLF-Container/Komponenten anlegen, wobei jeder der FLF-Container andere Füllzeichen haben kann.

Der Beispieltextbereich bietet eine Übersicht über das aktuelle FLF-Fragment und gestattet Ihnen, einzelne Feldnamen, Feldlängen und Breiten zu definieren. Jede Spalte wird in der Textkomponente in MapForce als mappbares Element dargestellt.

Die **Standardeinstellungen** des Containers sind:

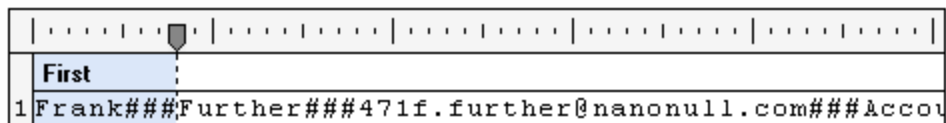
Füllzeichen (keines)
 Erste Zeile enthält Feldnamen nein
 Leere Felder als nicht vorhanden behandeln ja



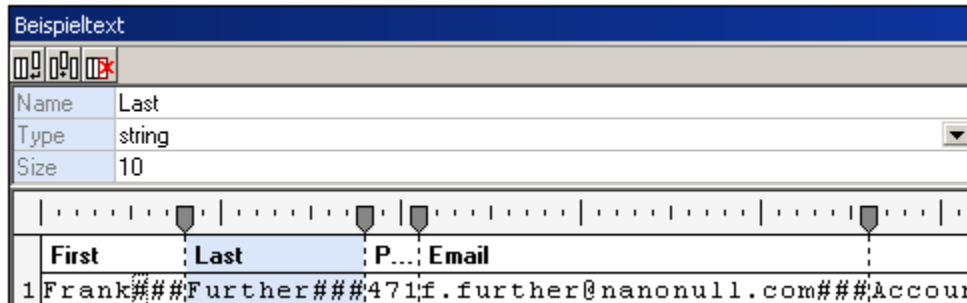
Konfigurieren des FLF-Containers/der Daten:

Nachdem Sie einen Container als "Speichern FLF" definiert haben, wird der Beispieltextbereich so wie im oben gezeigten Screenshot angezeigt. Es wird automatisch ein Standardfeld mit der Breite 10 eingefügt.

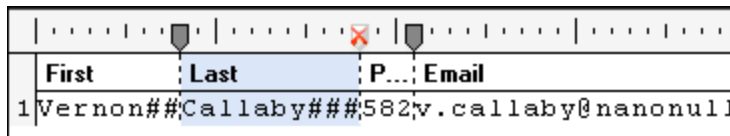
- Klicken Sie im Lineal auf das Tabulatorsymbol und ziehen Sie es, um es neu zu positionieren. Es erscheint ein Popup-Fenster mit der aktuellen Position.
- Wenn Sie den Cursor über das Lineal platzieren, wird ein "Punkt" angezeigt. Wenn Sie darauf klicken, wird an dieser Position ein neues Tabulatorzeichen eingefügt.



- Nachdem Sie die neue Position definiert haben, klicken Sie auf das Feld, um es auszuwählen und bearbeiten Sie den Namen im Feld "Name".



- Um ein Feld zu **entfernen**, klicken Sie auf das Tabulatorsymbol und ziehen Sie es aus dem Lineal. Nach erfolgreicher Ausführung dieser Aktion ändert sich das Tabulatorsymbol.



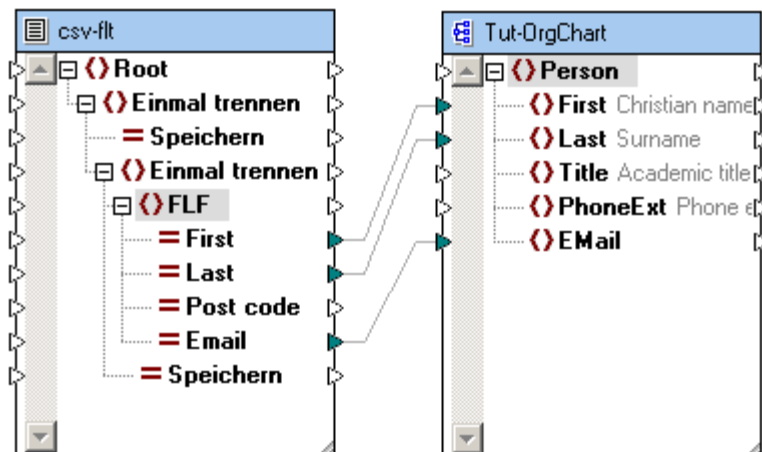
Bitte beachten Sie:

Wenn Sie einen Tabulator auf dem Lineal verschieben, werden automatisch alle Tabulatoren rechts davon verschoben. Um die anderen Tabulatorpositionen beizubehalten, halten Sie die Umschalttaste gedrückt, bevor Sie den Tabulator verschieben.

Wenn Sie im Beispieltext auf ein Feld klicken, erscheint dieses markiert, sodass Sie es näher konfigurieren können.

- Klicken Sie auf das Symbol "Anhängen" , um ein neues Feld mit der Länge 10 anzuhängen.
- Klicken Sie auf das Symbol "Einfügen" , um vor dem derzeit aktiven Feld ein Feld mit der Länge 10 einzufügen.
- Klicken Sie auf das Symbol "Löschen" , um das derzeit aktive Feld zu löschen.
- Klicken Sie in das Feld **Name**, um den hier erscheinenden Standardtext zu editieren.
- Klicken Sie in das Feld **Type**, um die Felddatentypen zu definieren: string, boolean, decimal usw.

Wenn Sie die FlexText-Vorlage in MapForce einfügen, können Sie die Daten auf jede der unterstützten Zieldateien mappen. In diesem Beispiel werden FLF-Elemente auf XML-Elemente gemappt.

**Leere Felder als nicht vorhanden behandeln**

Damit können Sie definieren, dass für leere Felder in der Quelldatei keine entsprechenden leeren Datenelemente (Element oder Attribut) in der Zieldatei erzeugt werden.

Ein Feld wird als nicht vorhanden betrachtet, wenn zwischen zwei aufeinanderfolgenden Füllzeichen keine Daten vorhanden sind.

19.7.8 Als Wert speichern

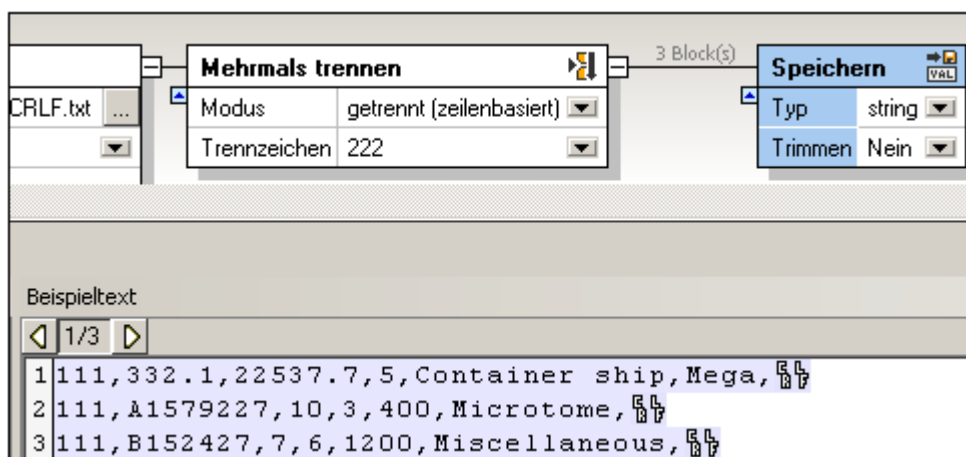


Mit dieser Option können Sie einen Container definieren, der die darin enthaltenen Daten in MapForce als zu mappendes Element zur Verfügung stellt. Wenn Sie den Namen des Containers in FlexText nicht ändern, wird das Mapping-Element mit dem Namen "Speichern" angezeigt.

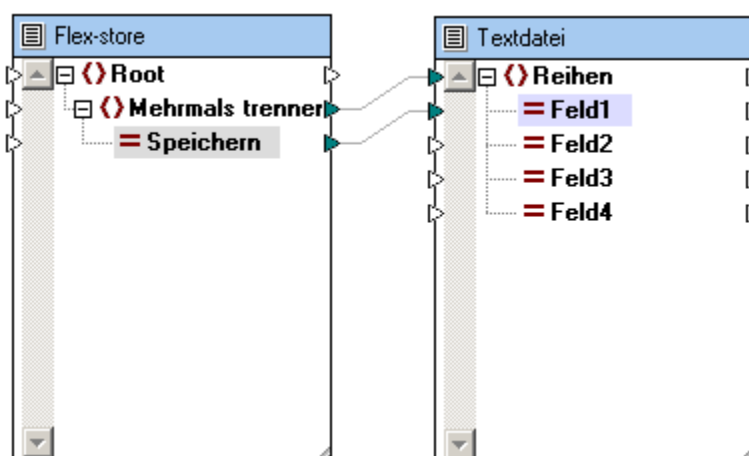
Die **Standardeinstellungen** des Containers sind:

Type string
Trimmen nein

Im unten gezeigten Screenshot sehen Sie den "Speichern"-Container sowie dessen Inhalt, der im Beispieltextbereich angezeigt wird.



Wenn Sie diese Vorlage speichern und in MapForce öffnen, können Sie das Element "Speichern" auf andere Elemente in einer Zielkomponente mappen.



Bitte beachten Sie:

Das Element Feld1 in der Zieldatei wird 3 Fragmente enthalten, die aus dem Element "Speichern" stammen, wenn Sie auf das Register "Ausgabe-Vorschau" klicken, um eine Vorschau des Ergebnisses zu sehen.

Type

Gestattet Ihnen, den Datentyp der Textfragmente zu definieren.

Trimmen

Definiert die Seite, von der die Zeichen getrimmt, also gekürzt werden sollen: links, rechts, oder beide. Bei Auswahl von "Ja" wird die Option "Zeichenfolge" aktiviert.

Zeichenfolge

Definieren Sie die Zeichen, die Sie von diesem Textfragment trimmen möchten. Sie können beliebig viele Zeichen eingeben, indem Sie in das Feld doppelklicken. Die Zeichen, die Sie hier eingeben, werden von der Trimm-Seite/den Trimm-Seiten des Fragments entfernt.

Kapitel 20

EDI - UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc

20 EDI - UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc

Altova Website:  [Mappen von EDI- / EDIFACT- / HL7-Dateien](#)

MapForce unterstützt eine Reihe von EDI (Electronic Data Interchange)-Formaten und ermöglicht das Mappen von Daten von und auf diese EDI-Formate. Beachten Sie bitte, dass Sie Java, C#, C++ oder BUILTIN (den Built-in-Ausführungsprozessor) als Ausgabesprache wählen müssen, um mit EDI-Dateien arbeiten zu können.

Auf der Altova-Website stehen für die einzelnen EDI-Formate separate Installer zur Verfügung. Zum Herunterladen dieser Konfigurationsdateien besuchen Sie bitte die [MapForce-Komponentenseite](#).

Die in der MapForce-Installationsdatei enthaltenen Konfigurationsdateien werden normalerweise mit jeder neuen MapForce-Version aktualisiert. Wenn Sie eine bestimmte EDI-Standardversion benötigen, laden Sie die Konfigurationsdateien bitte von der oben erwähnten [MapForce-Komponentenseite](#) herunter.

EDI-Komponenten können in MapForce als Quell- und Zielkomponente verwendet werden. Diese Daten können auf beliebig viele XML-Schemas, Datenbanken oder andere Komponenten gemappt werden.

MapForce kann nun mehrere Nachrichtentypen pro EDI-Komponente (einer einzigen Standardversion) verarbeiten. Dadurch können folgende Dateien verarbeitet werden:

- mehrere Dateien, die unterschiedliche Nachrichten als Input-Instanzen enthalten
- unterschiedliche Nachrichtentypen innerhalb derselben Datei, d.h. Datenaustausch

UN/EDIFACT

EDIFACT ist ein De-facto-Industriestandard (also ein UN-Standard) für den Dokumentenaustausch und MapForce unterstützt die in den Verzeichnissen **93A - 10A** des UN/EDIFACT-Standards enthaltenen Messages, wobei **2010A** als Standardverzeichnis verwendet wird. Dieses Verzeichnis enthält etwa 200 verschiedene Message Types. Nähere Informationen zu Verzeichnis-Downloads, Service Codes usw. finden Sie unter <http://www.unece.org/trade/untdid/welcome.htm>.

ANSI X12

ist ein Industriestandard für den Dokumentenaustausch und MapForce unterstützt die Versionen 3040, 3050, 3060, 3070, 4010, 4020, 4030, 4040, 4041, 4042, 4050, 4051, 4052, 4060, 5010, 5011, 5012, 5020, 5030, 5040, 5050, 6010 und 6020 wobei als Standard **ANSI X12**-Spezifikation **6020** verwendet wird. Informationen zu X12-Publikationen finden Sie unter <http://www.x12.org/>.

ANSI X12-Komponenten haben "virtuelle" Nodes, in die, je nachdem, welche Einstellungen Sie im Dialogfeld [EDI-Validierungseinstellungen](#) auswählen, EDI-Parser-Fehlerinformationen/Daten geschrieben werden. Ein X12 EDI 997 Functional Acknowledgement kann von jedem X12-Dokument generiert werden.

HL7

ist ein Industriestandard für den Austausch von Daten zwischen medizinischen Applikationen und steht für "Health Level Seven". MapForce unterstützt die Versionen 2.2 bis 2.6, wobei als Standard Version **2.6** installiert wird. Die XML-basierte Version HL7 Version 3.x wird in MapForce 2012 auf Basis von XML-Schemakomponenten unterstützt.

Auf der Seite [MapForce-Bibliotheken](#) der Altova-Website steht ein eigener Installer für die zusätzlichen Versionen HL7 V2.2 - V2.5.1 - und die HL7 V3.x XML-Schemas und Konfigurationsdateien zur Verfügung.

Beachten Sie, dass der Ordner **EDI and X12 mappings** des MapForceExamples-Projekts die **HL7V260_to_HL7V3.mfd** Beispieldatei enthält, in der eine HL7 V2.6 Datei auf eine HL7 V3 XML-Datei gemappt wurde.

Nähere Informationen zu HL7 finden Sie unter <http://www.hl7.org/>.

IATA PADIS

PADIS (Passenger and Airport Data Interchange Standards) ist ein Industriestandard für den Austausch von Passagier- und Flughafendaten mittels EDI-Dokumenten. MapForce unterstützt die Versionen 00.1 bis 08.1

Nähere Informationen zu PADIS finden Sie unter <http://www.iata.org/Pages/default.aspx>.

SAP IDocs

SAP IDocs (intermediate documents)-Dokumente dienen zum Austausch von Geschäftsdaten zwischen SAP R/3-Applikationen und Nicht-SAP-Applikationen. Bei den Dokumenten handelt es sich um ein Zwischenformat, das zwischen verschiedenen Systemen ausgetauscht werden kann.

Nähere Informationen zu SAP IDocs finden Sie unter: <http://sap.com>.

HIPAA X12

HIPAA basiert auf dem X12 EDI 5010-Standard, hat aber eigene spezialisierte Versionen, die nativ von MapForce 2011 Release 3 oder höher unterstützt werden. Es stehen zusätzliche Installer für HIPAA 5010A1 und 5010A2 zur Verfügung.

Herunterladen zusätzlicher Bibliotheken:

Besuchen Sie bitte die Seite [MapForce-Bibliotheken](#), um zusätzliche EDI-Konfigurationsdateien in ihren jeweiligen Installern herunterzuladen.

20.1 EDI-Terminologie

In der folgenden Liste werden die wichtigsten UN/EDIFACT-Begriffe und ihre Entsprechungen in ANSI X12, dem in den USA gebräuchlichen Standard beschrieben.

Messages (ANSI X12 - Transactions)

Ein Message (oder Transaction Set) ist eine bestimmte Sequenz von Segmenten, die ein Geschäftsdokument darstellt. MapForce unterstützt einen oder mehrere Message Types (Nachrichtentypen) in einer einzigen EDI-Komponente.

Segment

Ein einzelner "Datensatz" in einer Message.

Segmente werden durch eine aus zwei oder drei Zeichen bestehende ID am Anfang des Segments identifiziert. Eine Gruppe von verwandten Elementen wird durch einen Segment-Tag (bzw. Segment ID - ANSI X12) zusammengefasst. Segmente einer Transaktion können als Muss-Angabe (engl. mandatory) oder Kann-Angabe (engl. optional) definiert werden.

Element

Ein einzelnes Datenfeld innerhalb eines Segments.

Ein Element ist als Feld zu verstehen, d.h. es enthält einen Datentyp, einen Namen oder eine Adresse, z.B. Elemente lassen sich weiter in Composite Elements bestehend aus Component Elements oder Unterelementen unterteilen.

Trennzeichen

Elemente werden durch "Trennzeichen" voneinander getrennt. In UN/EDIFACT werden hierfür entweder Standardzeichen verwendet, oder die Trennzeichen werden in einem optionalen UNA-Kontrollsegment definiert.

Standardzeichen

Doppelpunkt	:	Trennzeichen für Component Elements
Plus	+	Trennzeichen für Datenelemente
Hochkomma	'	Segment-Endezeichen

HL7 unterstützt ein zusätzliches Trennzeichen für ein untergeordnetes Feld, wobei das Standardzeichen & ist. Dieses Trennzeichen wird im Dialogfeld "Komponenteneinstellungen" als [Unterkomponenten-Trennzeichen](#) bezeichnet.

Interchange

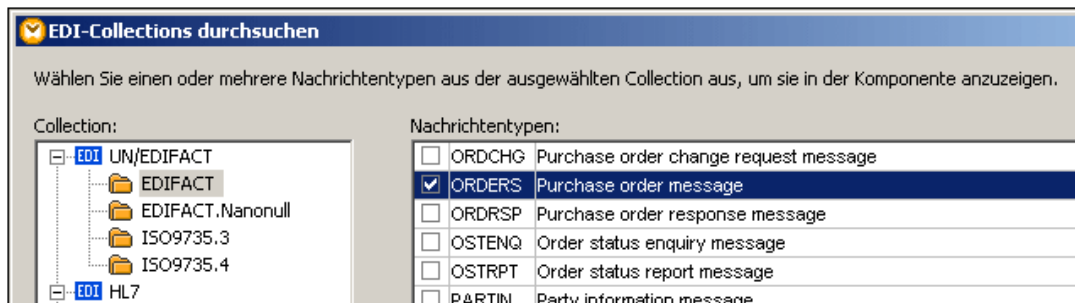
Eine Sammlung von Group Envelopes und/oder Messages an denselben Empfänger.

20.2 Mappen von UN/EDIFACT auf ein XML-Schema

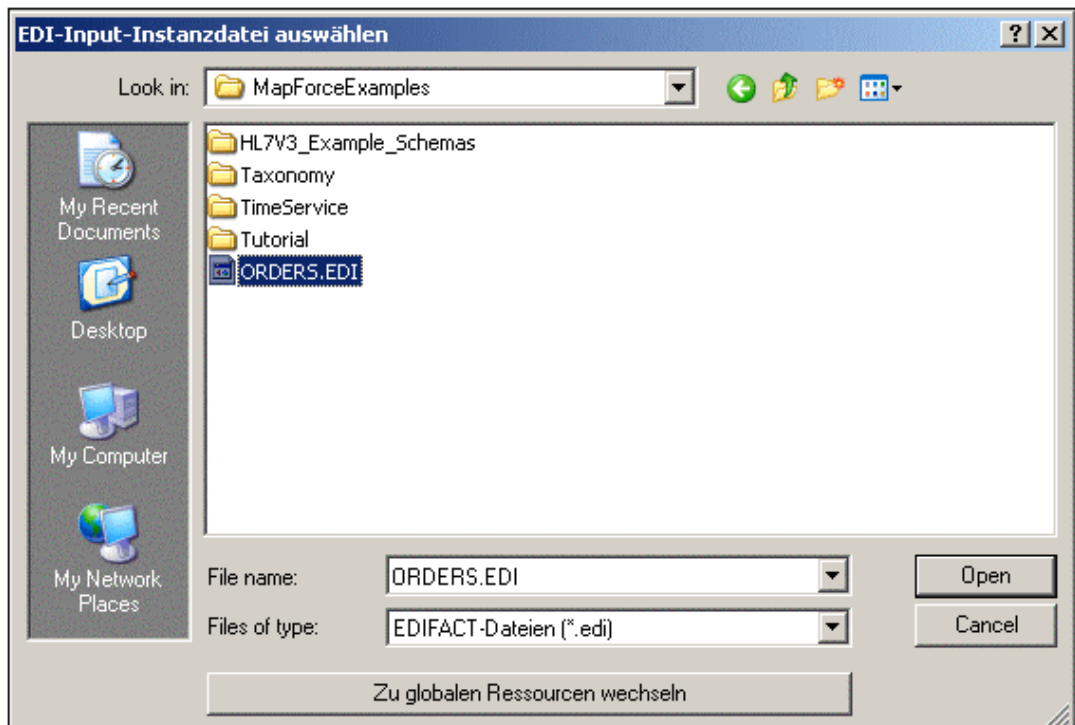
Ziel dieser Übung ist zu zeigen, wie Daten von UN/EDIFACT-Messages auf ein XML-Schema/Dokument gemappt werden, um eine XML-Datei zur weiteren Verarbeitung zu erzeugen. Sie finden das in diesem Beispiel erläuterte Mapping im Verzeichnis **...\\MapForceExamples** unter dem Namen **EDI_Order.mfd**.

Erzeugen der EDIFACT-Komponente in MapForce:

1. Erstellen Sie ein neues Projekt und stellen Sie sicher, dass entweder Java, C#, C++ oder BUILTIN aktiv ist. XSLT 1.0- / 2.0- oder XQuery-Code kann beim Mappen von EDI-Dateien nicht generiert werden.
2. Wählen Sie die Menüoption **Einfügen | EDI** oder klicken Sie auf das EDI-Symbol und wählen Sie aus der Liste der EDI-Collections den Eintrag **ORDERS** aus und klicken Sie auf OK.



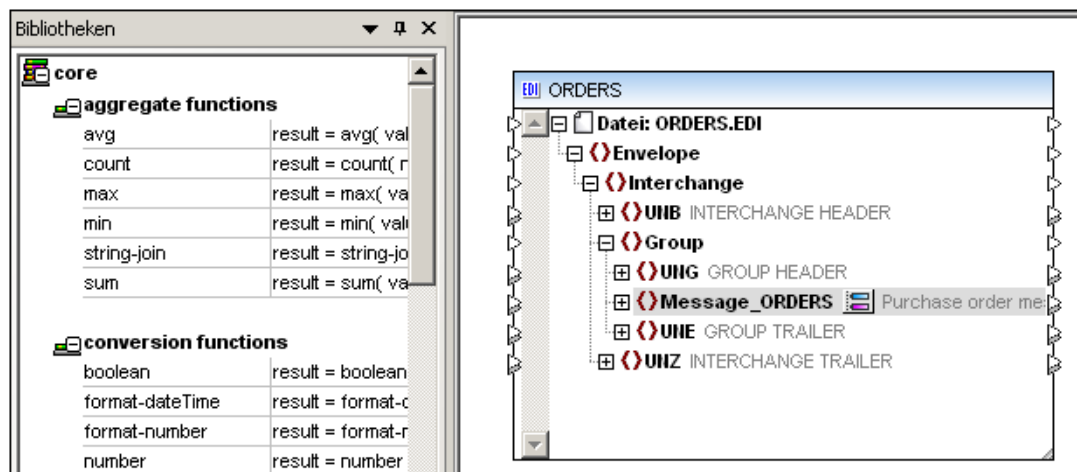
3. Klicken Sie auf die Schaltfläche "Durchsuchen" und wählen Sie im Verzeichnis **...\\MapForceExamples** die Datei **ORDERS.EDI** aus und klicken Sie auf "Öffnen".



Daraufhin wird das Dialogfeld "Komponenteneinstellungen" geöffnet, indem Sie die

Input- und Output-Edi-Instanz auswählen können. Klicken Sie zum Fortfahren auf Weiter.

4. Die EDI-Komponente wird nun im Mapping-Fenster angezeigt.



Java ausgewählt

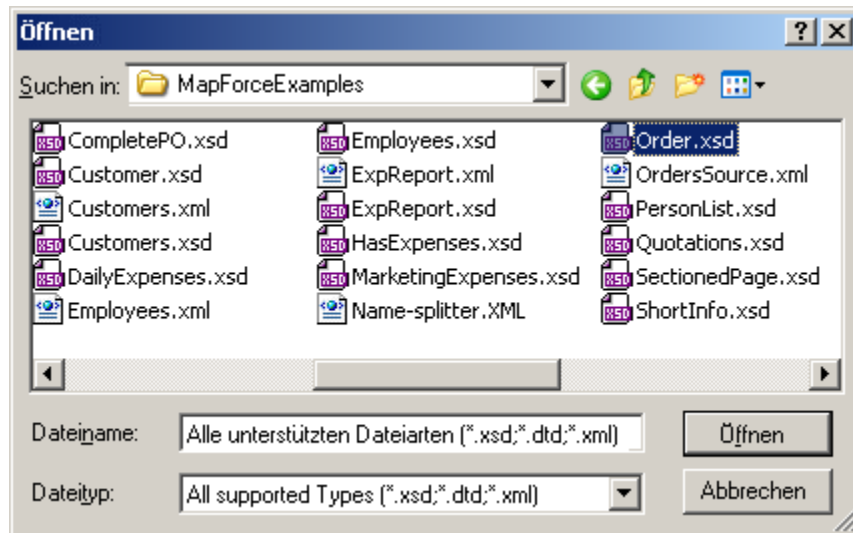
5. Klicken Sie auf den Eintrag **Message_ORDERS** und drücken Sie auf der Zahlentastatur die *-Taste, um alle Elemente anzusehen.
6. Klicken Sie auf das **Vergrößerungssymbol** in der rechten unteren Ecke des Komponentenfensters und passen Sie die Größe des Fensters an.



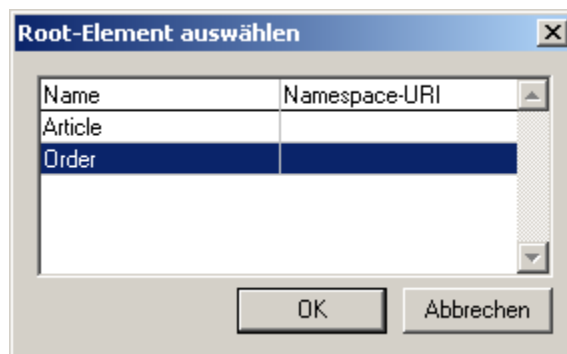
Java ausgewählt

Erzeugen der Zielschemakomponente:

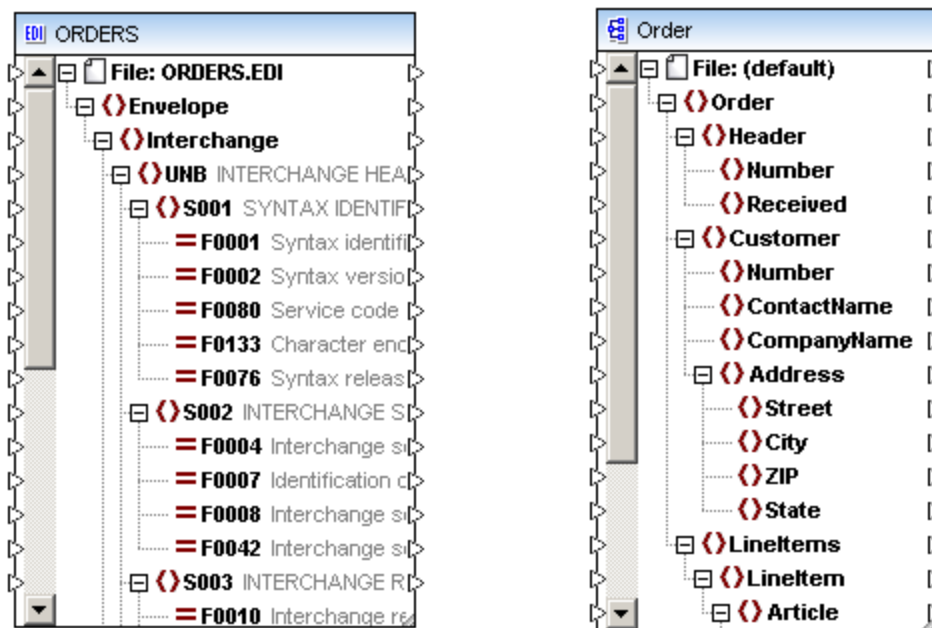
1. Klicken Sie auf das Symbol **XML-Schema/Datei einfügen** und wählen Sie im Verzeichnis ...**MapForceExamples** die Datei **Order.xsd** aus.



2. Wenn Sie aufgefordert werden, eine XML-Beispieldatei anzugeben, klicken Sie auf **Überspringen** und wählen Sie **Order** als Root des Zieldokuments aus.



3. Klicken Sie auf den Eintrag **Order** und drücken Sie die *-Taste auf der Zahlentastatur, um alle Elemente anzusehen.
4. Klicken Sie auf das **Vergrößerungssymbol** in der rechten unteren Ecke des Komponentenfensters und passen Sie die Fenstergröße an.



Wir können nun mit dem Mappen von EDI-Elementen von der EDI-Quellkomponente auf das Zielschema beginnen.

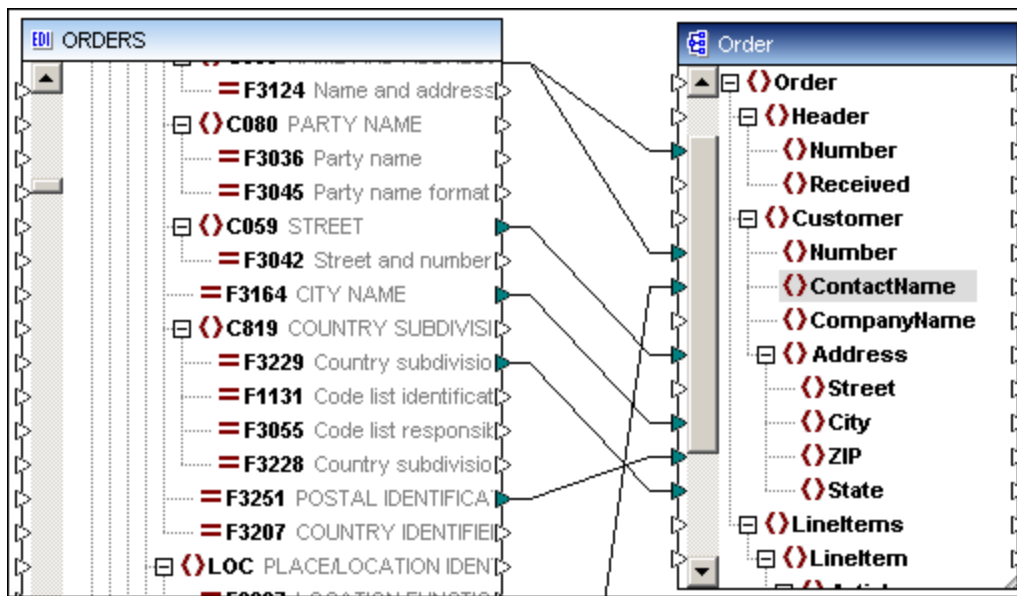
Mappen der EDI-Datenelemente:

In der EDI-Komponente wird nun die Struktur einer Message angezeigt. Diese Struktur basiert auf der soeben ausgewählten Collection (ORDERS).

Normalerweise enthalten nicht alle Nodes Daten, daher muss der Projektautor das jeweils verarbeitete EDI-Dokument gut genug kennen, um die relevanten Nodes zu finden. In diesem Fall müssen die folgenden Nodes (beginnend mit dem Group/Message Node) direkt gemappt werden:

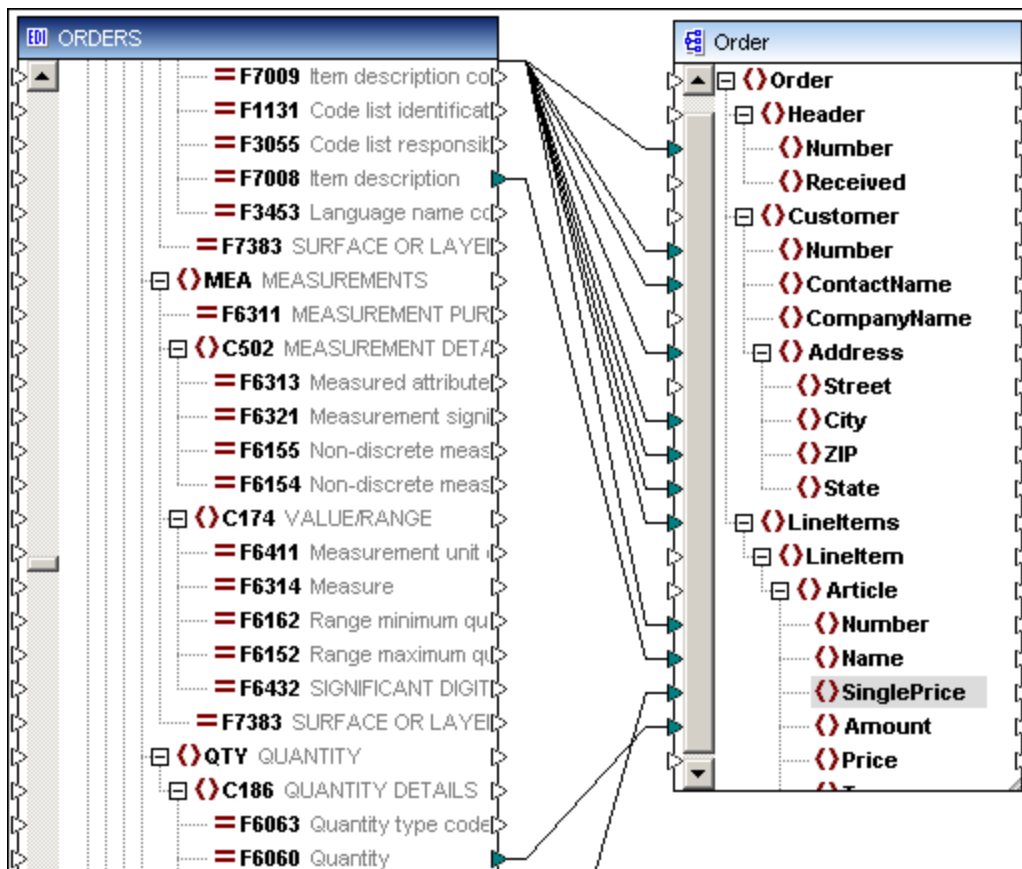
BGM/C106/F1004	-> Order/Header/Number
SG2/NAD/C082/F3055	-> Order/Customer/Number
SG2/NAD/C080/F3036	-> Order/Customer/CompanyName
SG2/NAD/C059/F3042	-> Order/Customer/Address/Street
SG2/NAD/F3164	-> Order/Customer/Address/City
SG2/NAD/C819/F3229	-> Order/Customer/Address/State
SG2/NAD/F3251	-> Order/Customer/Address/ZIP
SG2/SG5/CTA/C056/F3412	-> Order/Customer/ContactName

In dieser Phase sollte das Mapping ähnlich wie in der unten gezeigten Abbildung aussehen:



Fahren Sie mit dem Mapping fort und mappen Sie:

SG28	->Order/LineItems
SG28/LIN/C212/F7140	->Order/LineItems/LineItem/Article/Number
SG28/IMD/C273/F7008	->Order/LineItems/LineItem/Article/Name
SG28/QTY/C186/F6060	->Order/LineItems/LineItem/Article/Amount
SG28/SG32/PRI/C509/F5118	->Order/LineItems/LineItem/Article/SinglePrice

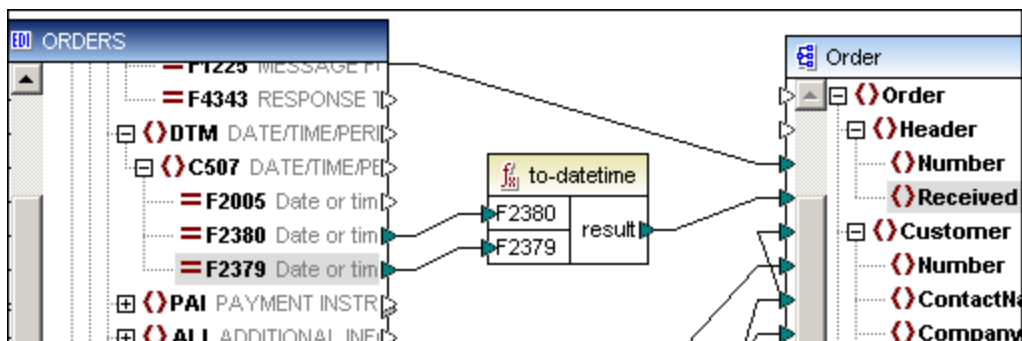


Ziehen Sie eine **to-datetime**-Funktion aus der **edifact**-Bibliothek in das Mapping-Fenster.

Durch Anwendung der Komponenten F2380 und F2379 des Elements **DTM/C507** lässt sich ein ordnungsgemäß formatiertes datetime-Element **Received** erzeugen.

Wir mappen daher die folgenden Felder:

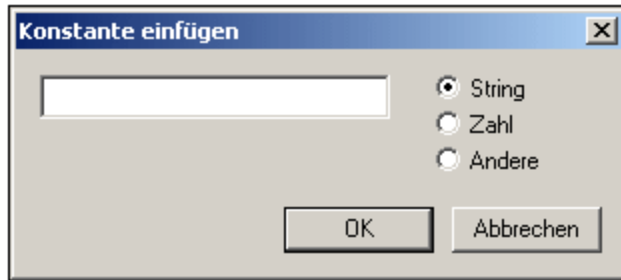
DTM/C507/F2380	-> die F2380-Eingabe der to-datetime-Funktion
DTM/C507/F2379	-> die F2379-Eingabe der to-datetime-Funktion
result der to-datetime -Funktion	-> Order/Header/Received



Filtern der Bestellungen des Käufers (Buyer):

Wir wollen nun die Bestellungen des Käufers (Buyer) filtern. Diese lassen sich anhand des "party function code qualifier" des NAD (Name and Address)-Segments identifizieren. In diesem Fall bedeutet der Wert 'BY' einen Buyer, also Käufer (Partei an den eine Ware und/oder Dienstleistung verkauft wird).

1. Ziehen Sie eine **equal**-Funktion aus der **core**-Bibliothek in das Mapping-Fenster.
2. Wählen Sie die Menüoption **Einfügen | Filter: Nodes/Zeilen**.
3. Fügen Sie eine Konstantenkomponente ein. Wählen Sie dazu die Menüoption **Einfügen | Konstante**, klicken Sie auf "String" und geben Sie im Textfeld 'BY' ein:



Mappen Sie die folgenden Datenelemente:

SG2/NAD/F3035

-> der **b**-Eingabewert der **equal**-Funktion

Konstante "BY"

-> der **a**-Eingabewert der **equal**-Funktion

result der **equal**-Funktion

-> der **bool**-Eingabewert der **filter**-Funktion

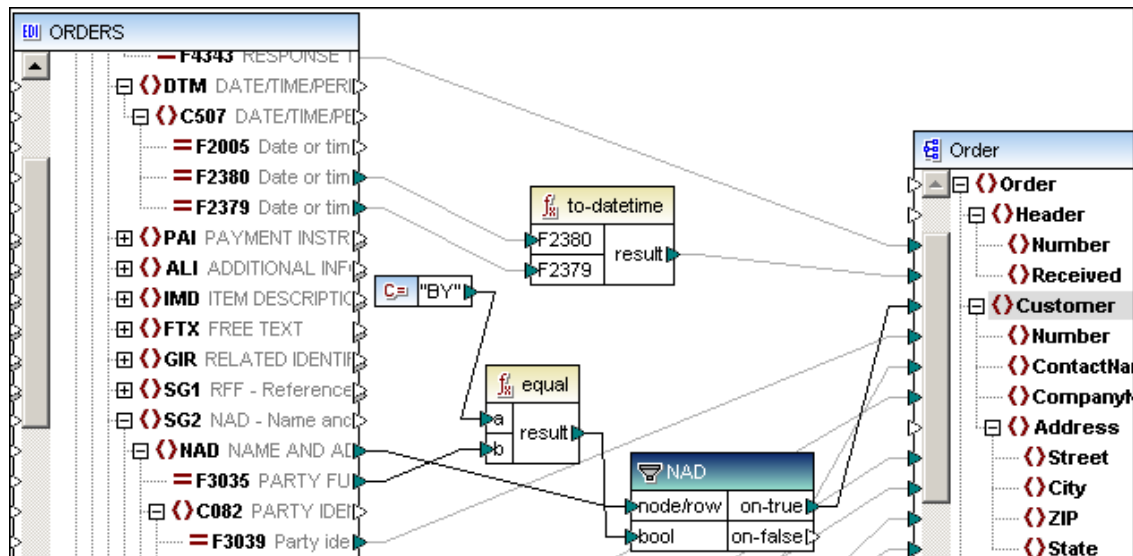
SG2/NAD

-> der **node/row**-Eingabewert der **filter**-Funktion

result der **filter**-Funktion

-> **Order/Customer** im Schema

Ziel ist, nur Daten zu mappen, deren NAD auf einen 'Buyer' referenziert, wie vom "party function code qualifier" 'BY' gekennzeichnet.



Der letzte Schritt in diesem Vorgang ist nun die Berechnung des Preises und der Steuern.

1. Ziehen Sie aus der **core**-Bibliothek zwei **multiply**- und eine **divide**-Funktion in das Mapping-Fenster.
2. Fügen Sie eine Konstantenkomponente ein (**Einfügen | Konstante**), stellen Sie sicher, dass "Numerisch" ausgewählt ist und geben Sie im Textfeld 100.0 ein.
3. Mappen Sie die folgenden Elemente:

SG28/QTY/C186/F6060

-> **value1** der ersten **multiply**-Funktion

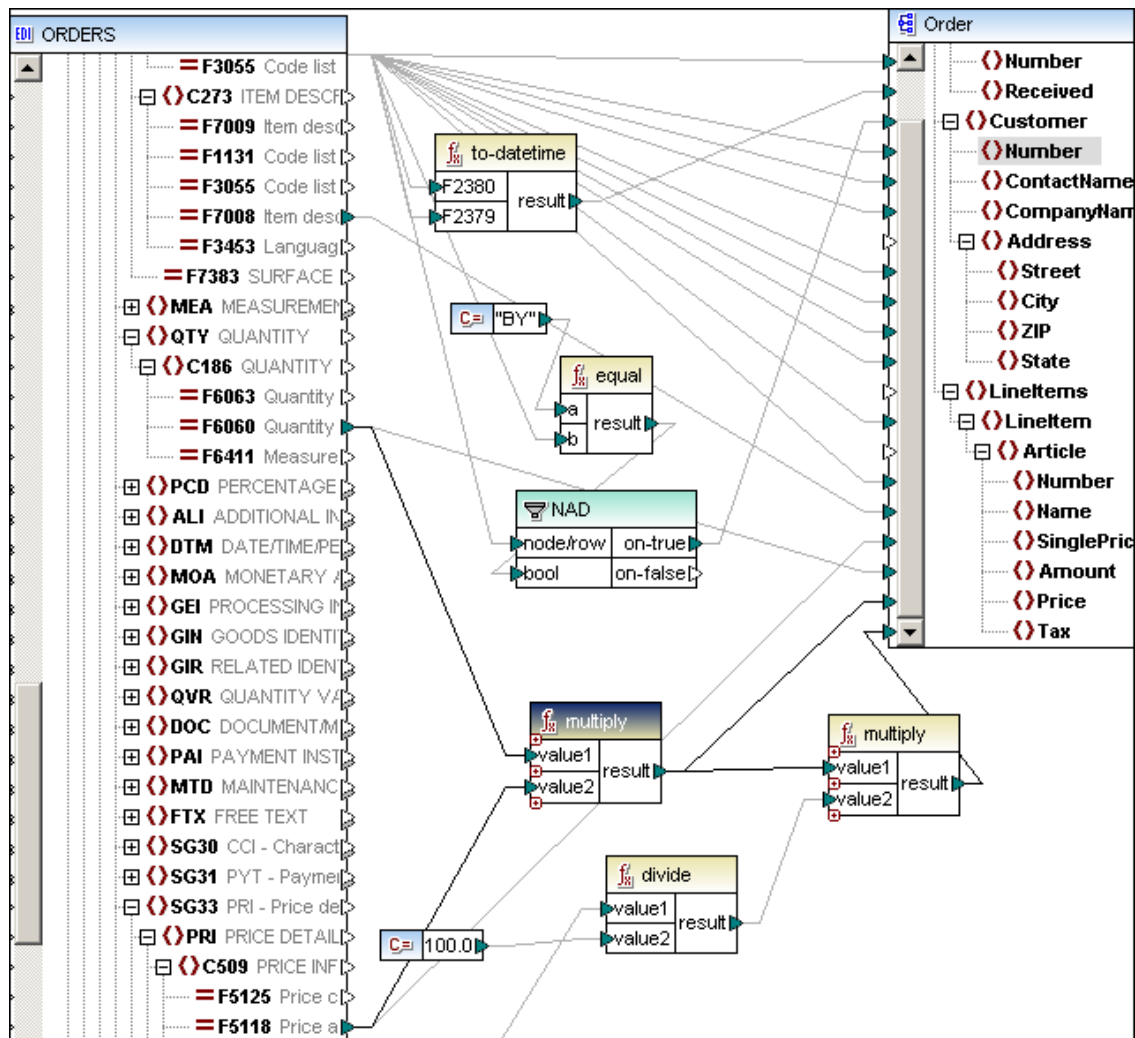
SG28/SG32/PRI/C509/F5118
result der ersten **multiply**-Funktion
 SG28/SG38/TAX/C243/F5278
 Die **Konstante** "100.0"

-> **value2** der ersten **multiply**-Funktion
 -> Order/LineItems/LineItem/Article/Price
 -> **value1** der **divide**-Funktion
 -> **value2** der **divide**-Funktion

result der ersten **multiply**-Funktion
result der **divide**-Funktion
result der zweiten **multiply**-Funktion

-> **value1** der zweiten **multiply**-Funktion
 -> **value2** der zweiten **multiply**-Funktion
 -> Order/LineItems/LineItem/Article/Tax

Ihr Mapping sollte nun folgendermaßen aussehen:



Wenn Sie auf die Schaltfläche Ausgabe klicken, wird eine on-the-fly-Transformation durchgeführt und das erzeugte XML-Dokument angezeigt:



20.3 Validierung von EDI-Komponenten

MapForce kann all unterstützten EDI-**Quell-** und **-Zieldokumente** validieren, wenn das Mapping ausgeführt wird, z.B. durch Klicken auf die Schaltfläche "Ausgabe". Bei der Validierung handelt es sich jedoch nicht um eine komplette Validierung der EDI-Syntax und -Semantik. In der Projektdatei **X12_To_XML_Order.mfd** im Ordner ...MapForceExamples sehen Sie ein Beispiel für diese Art von Mapping.

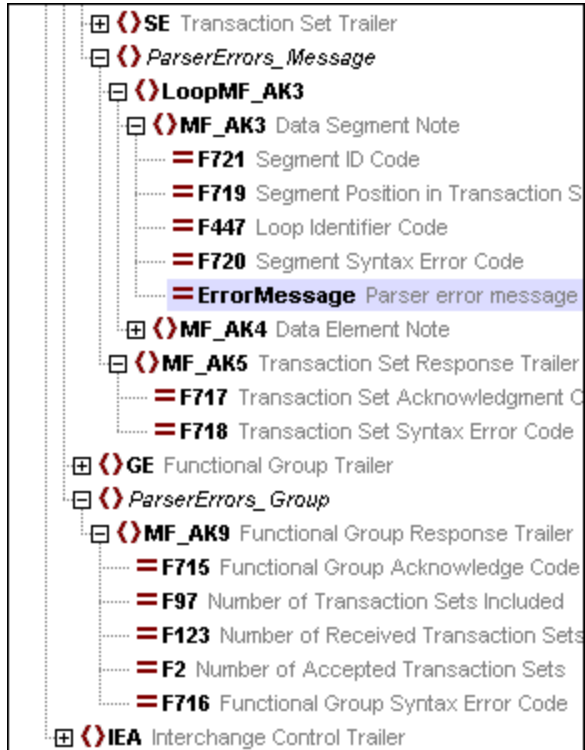
Die Validierungsparameter können vom Benutzer je nach Bedarf konfiguriert werden. Sie können festlegen, dass Fehler ignoriert werden, der Validierungsvorgang bei einem bestimmten Fehler gestoppt wird oder dass Fehler akzeptiert/zurückgewiesen oder gemeldet werden.

Bei Ausführung des generierten Programmcodes lösen Parsing-Fehler ein Ausnahmeereignis aus, wodurch das Mapping gestoppt wird und aufgrund der Validierungsfehler wird eine entsprechende Meldung in der Standardausgabe angezeigt. Dies kann durch Bearbeitung der SPL-Vorlagen (des generierten Codes) oder durch Ändern der Werte für ErrorMask und WarningMask behoben werden.

X12 Acknowledgment

Die Validierungsergebnisse (der Validierungsbericht) für EDI-Quellkomponenten werden in "virtuelle" Datenelemente am unteren Rand der EDI-Komponente (unter **ParserError_Message** und **ParserErrors_Group**) platziert. Diese Ergebnisse können bei der Generierung eines X12 997 oder 999 Functional Acknowledgment verwendet werden.

Informationen zur Erstellung einer X12 Acknowledgment-Datei, die über den Status des Datenaustauschs Auskunft gibt, finden Sie im Abschnitt [X12 997 - Functional Acknowledgment](#) oder [X12 999 - Implementation Acknowledgment](#).



So passen Sie die Validierungseinstellungen an:

1. Doppelklicken Sie auf eine EDI-Komponente und klicken Sie im Dialogfeld

"Komponenteneinstellungen" auf die Schaltfläche **Validierung....**

In der Abbildung unten sehen Sie die Standardeinstellungen.

2. Wählen Sie die jeweiligen Validierungseinstellungen aus und klicken Sie zur Bestätigung auf OK.

	Stopp	Melden & Ablehnen	Melden & Akzeptieren	Ignorieren
Fehlendes Segment	☒	☐	☐	☐
Nicht erwartetes Segment	☒	☐	☐	☐
Nicht erkannte Segment-ID	☒	☐	☐	☐
Fehlende Gruppe	☒	☐	☐	☐
Unerwartetes Dateiende	☒	☐	☐	☐
Fehlendes Feld oder Kompositum	☐	☒	☐	☐
Zusätzliche Daten in Segment oder Kompositum	☐	☒	☐	☐
Zusätzliche Wiederholung	☐	☒	☐	☐
Ungültiger Feldwert	☐	☒	☐	☐
Ungültiges Datum	☐	☒	☐	☐
Ungültige Uhrzeit	☐	☒	☐	☐
Numerischer Überlauf	☐	☒	☐	☐
Datenelement ist zu kurz	☐	☒	☐	☐
Datenelement ist zu lange	☐	☒	☐	☐
Ungültiger Codelistenwert	☐	☒	☐	☐
Semantischer Fehler	☐	☒	☐	☐

"Stopp" bricht den Vorgang bei einem Fehler sofort ab. In der Liste der Parserfehler wird keine Fehlermeldung angezeigt.

Bei **Stopp** werden schwerwiegende Fehler ausfindig gemacht und die Ausführung des Mappings abgebrochen, ohne dass ein Meldungsbericht generiert wird.

Bei **Melden & ablehnen** und **Melden & akzeptieren** werden in den Datenelementen Parser_Errors_Message und Parser_Errors_Group der EDI-Komponente Informationen ausgegeben. Diese Datenelemente können auf andere gemappt werden.

Je nach den Einstellungen für **Ablehnen** oder **Akzeptieren** enthalten der Functional Group Acknowledge Code F715 and the Transaction Set Acknowledgment Code F717 entweder:
den Wert 'R' für '*Rejected*' (abgelehnt) oder
den Wert 'E' für '*Akzeptiert, doch wurden Fehler festgestellt*'. Die Fehler werden auch im Fenster "Meldungen" angezeigt.

Bei **Ignorieren** wird der jeweilige Fehler ignoriert. In den Datenelementen Parser_Errors_Message und Parser_Errors_Group items werden in diesem Fall keine Informationen ausgegeben.

Im Folgenden werden die einzelnen Fehlermeldungen näher erläutert:

Name	Beschreibung	Fehler kann auftreten bei einer
Fehlendes Segment	Ein obligatorisches Segment fehlt oder kommt weniger oft als im Mindestwert definiert vor.	Quellkomponente
Nicht erwartetes Segment	Ein Segment ist zwar in der Spezifikation, nicht aber in dieser Nachricht definiert.	Quellkomponente
Nicht erkannte Segment-ID	Es wurde ein Segment gefunden, das in der Spezifikation nicht definiert ist.	Quellkomponente
Fehlende Gruppe	Eine obligatorische Gruppe fehlt oder kommt weniger oft als im Mindestwert definiert vor.	Quellkomponente
Unerwartetes Dateieinde	Die Instanz kann nicht geparkt werden, da einige Daten fehlen.	Quellkomponente
Fehlendes Feld oder Kompositum	Ein obligatorisches Feld oder Kompositum fehlt oder kommt weniger oft als im Mindestwert definiert vor.	Quell- oder Zielkomponente
Zusätzliche Daten in Segment oder Kompositum	Die Input-Instanz enthält zusätzliche Daten, die von der Syntaxbeschreibung nicht erwartet werden.	Quellkomponente
Zusätzliche Wiederholung	Die tatsächliche Anzahl der Felder in einem Segment/Kompositum übersteigt die definierte Maximalanzahl	Quell- oder Zielkomponente
Ungültiger Feldwert	Ein numerisches Feld enthält ein ungültiges Zeichen.	Quellkomponente
Ungültiges Datum	Ein Datumsfeld enthält entweder ein ungültiges Zeichen oder die Werte für den Monat oder Tag sind ungültig.	Quellkomponente
Ungültige Uhrzeit	Ein Uhrzeitfeld enthält entweder ein ungültiges Zeichen oder die Werte für die Stunden oder Minuten sind ungültig.	Quellkomponente
Numerischer Überlauf	Ein numerischer Wert geht über die definierte Domain hinaus. Dieser Fehler wird nur innerhalb des generierten Codes unterstützt.	Quellkomponente
Datenelement ist zu kurz	Die Länge eines Datenelemente ist geringer als der definierte Mindestwert.	Quell- oder Zielkomponente
Datenelement ist zu lange	Die Länge eines Datenelemente ist größer als der definierte Maximalwert.	Quell- oder Zielkomponente
Ungültiger Codelistenwert	Ein Codelistenwert stimmt nicht mit der Gruppe der definierten Werte überein.	Quell- oder Zielkomponente
Semantischer Fehler	Es ist ein semantischer Fehler aufgetreten.	Quell- oder Zielkomponente
Das Implementation "Not Used"-Datenelement ist vorhanden		

Die Input-Datei wurde nicht vollständig geparst		

Nachdem Sie die Datenelemente auf die Zielkomponente gemappt haben, klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

Wenn es Fehler im EDI-Quell- oder Zieldokument gibt, werden diese bei Ausführung des Mappings im Fenster "Meldungen" angezeigt. Bei einem Fehler in der EDI-Quellkomponente muss das Mapping nicht automatisch fehlschlagen. Unter Umständen wird dennoch eine Ausgabe auf dem Register "Ausgabe" generiert.

20.4 X12 997 - Functional Acknowledgment

Die Datei X12 997 Functional Acknowledgement gibt Auskunft über den Status des EDI-Datenaustauschs. Sie enthält eine Liste aller Fehler, die bei der Verarbeitung des Dokuments aufgetreten sind. MapForce kann automatisch eine X12 997-Komponente im Hauptmappingbereich erstellen und die nötigen Konnektoren automatisch verbinden.

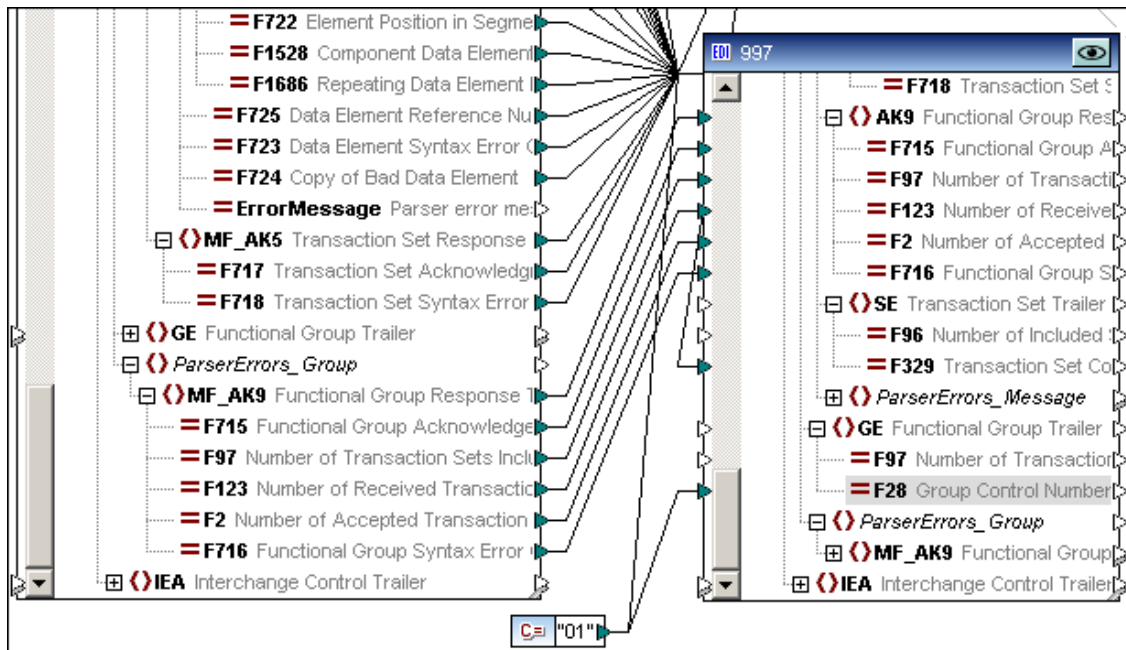
In der Projektdatei **X12_To_XML_Order.mfd** im Ordner ...MapForceExamples sehen Sie ein Beispiel für ein Mapping dieser Art.

Nähere Informationen zu den einzelnen Fehlersegmenten und ihrer Bedeutung finden Sie im Abschnitt [X12 997 Acknowledgment](#).

Beachten Sie, dass Sie statt dessen auch ein [X12 999 - Implementation Acknowledgment](#) generieren können.

So generieren Sie das EDI 997 Functional Acknowledgment:

1. Klicken Sie mit der rechten Maustaste auf die EDI-Quellkomponente und wählen Sie den Befehl "Mapping auf EDI X12 997 erstellen".



Daraufhin wird eine EDI 997-Komponente erstellt und die für die Generierung der X12 997-Bestätigung benötigten Datenelemente werden automatisch generiert.

2. Klicken Sie auf die Vorschau-Schaltfläche der EDI 997-Komponente und anschließend auf das Ausgaberegister, um die generierte Acknowledgment-Datei zu sehen.

1	ISA+00+ +00+ +ZZ+ReceiverID +ZZ+SenderID +100128+1113+!+00505+000000000+1+P+:'
2	GS+01+ReceiverID+SenderID+20100128+11130544+1+X+05012'
3	ST+997+0001'
4	AK1+PO+1+05012'
5	AK2+850+12345'
6	AK5+A'
7	AK9+A+1+1+1'
8	SE+6+0001'
9	GE+1+1'
10	IEA+1+000000000'
11	

So speichern Sie die 997 ACK-Datei:

- Geben Sie entweder im Dialogfeld "Komponenteneinstellungen" einen Dateinamen in das Feld "EDI Output-Datei " ein oder
- klicken Sie, wenn Sie Code generieren und das Ergebnis im Ausgabefenster anzeigen, auf die Schaltfläche "Generierte Ausgabe speichern"  und geben Sie einen Pfad- und Dateinamen an.

20.5 X12 999 - Implementation Acknowledgment

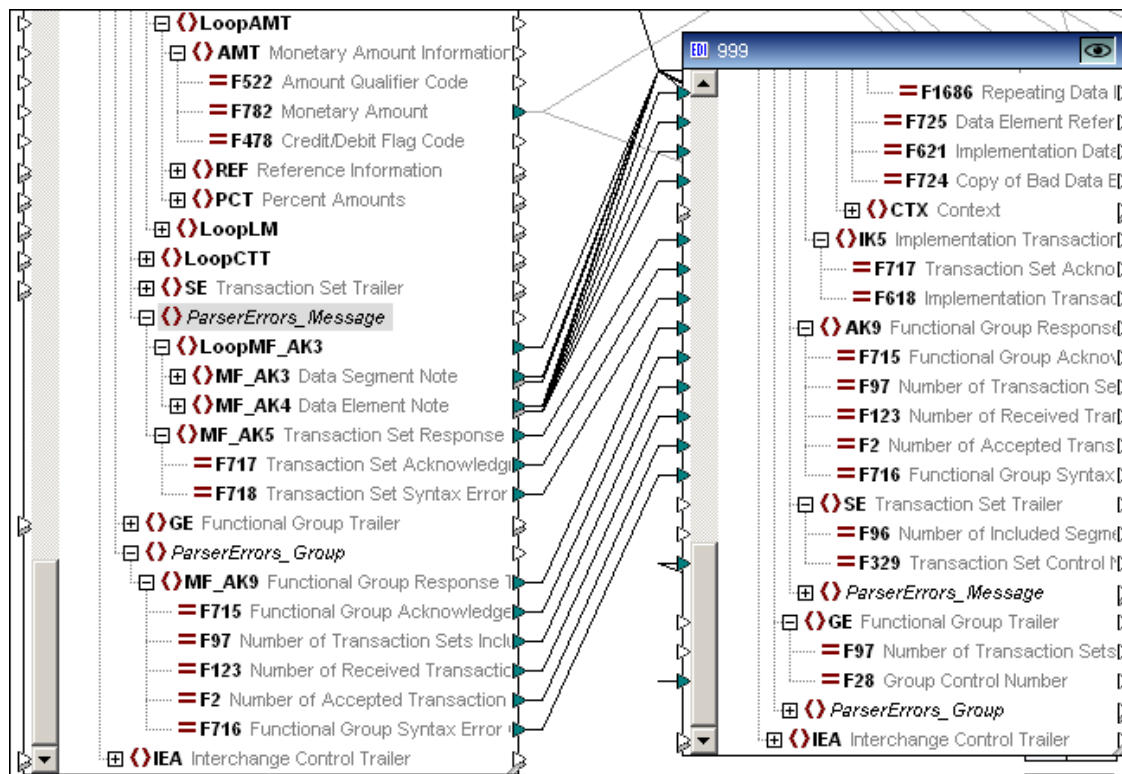
Das X12 999 Implementation Acknowledgment Transaction Set meldet die Nichterfüllung des X12 Implementation Guide oder Applikationsfehler.

Da es sich hierbei um eine Obermenge des 997 Functional Acknowledgement handelt, kann 999 anstelle von 997 verwendet werden, um Transaction Sets auf Basis entweder der Syntaxanforderungen der X12- oder der HIPAA-Implementierungsrichtlinien anzunehmen oder zurückzuweisen.

Alle Fehler, die bei der Verarbeitung des Dokuments auftreten, werden in Form einer Meldung ausgegeben, z.B. "Required Segment Missing", "Required Data Element Missing", "Code Value Not Used in Implementation", usw. MapForce kann automatisch eine X12 999-Komponente im Bereich des Hauptmappings generieren und die nötigen Konnektoren automatisch verbinden.

So generieren Sie das EDI 999 Implementation Acknowledgment:

1. Klicken Sie mit der rechten Maustaste auf die EDI-Quellkomponente und wählen Sie den Befehl "Mapping auf EDI X12 999 erstellen".



Daraufhin wird eine EDI 999-Komponente erstellt und die zum Generieren des X12 999 Implementation Acknowledgment benötigten Datenelemente werden automatisch verbunden.

2. Klicken Sie auf die Vorschau-Schaltfläche der EDI 999-Komponente und anschließend auf das Ausgaberegister um die generierte Acknowledgment-Datei zu sehen.

1	ISA+00+ +00+ +ZZ+ReceiverID +ZZ+SenderID +110502+1533+1+00505+000000000+1+P+:'
2	GS+FA+ReceiverID+SenderID+20110502+15333751+1+X+006020'
3	ST+999+0001'
4	AK1+PO+1+006020'
5	AK2+850+12345'
6	IK5+A'
7	AK9+A+1+1+1'
8	SE+6+0001'
9	GE+1+1'
10	IEA+1+000000000'
11	

So speichern Sie die 999 ACK-Datei:

- Geben Sie im Dialogfeld "Komponenteneinstellungen" entweder den Dateinamen in das Feld "EDI Output-Datei" ein, wenn Sie Code generieren, oder
- Klicken Sie auf die Schaltfläche "Generierte Ausgabe speichern" , wenn das Ergebnis im Ausgabefenster angezeigt wird, und geben Sie den Pfad und Dateinamen ein.

20.6 TA1-Segment

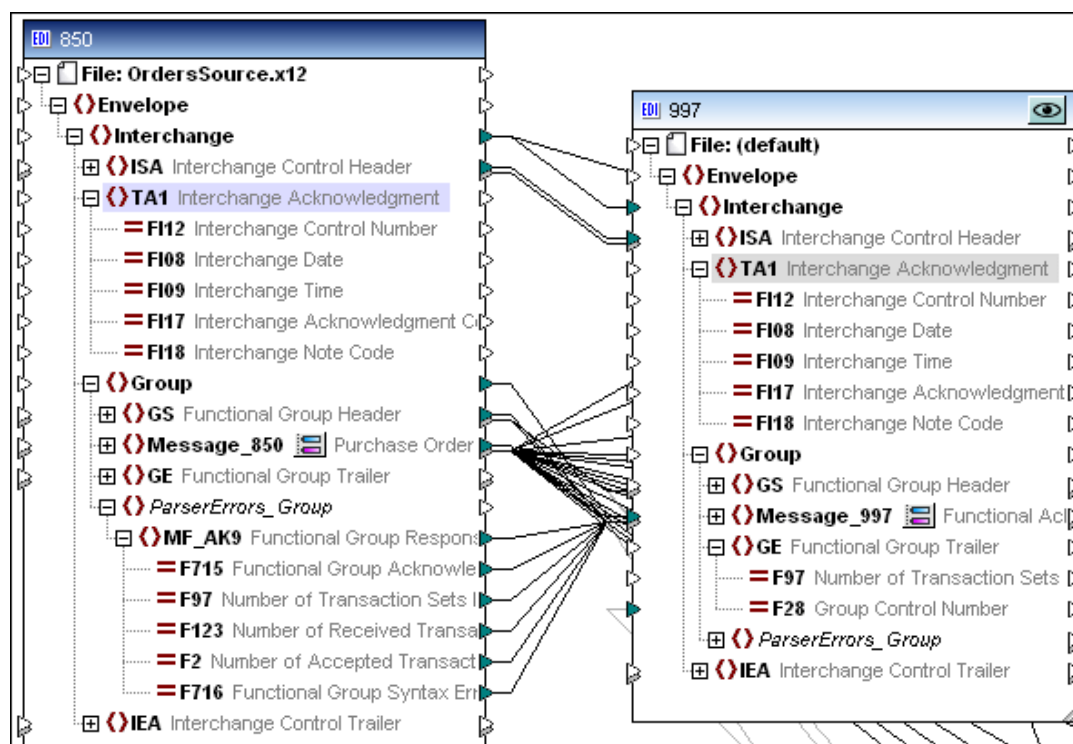
Das TA1-Segment ist ein optionales Segment, mit dem der Empfang des Interchange-Elements und die syntaktische Korrektheit der darin enthaltenen Envelope-Segmente bestätigt wird. Das Segment kann zu EDI X12- und EDI HIPAA-Komponenten hinzugefügt werden.

So inkludieren Sie das TA1-Segment in eine EDI-Komponente:

1. Navigieren Sie zum Ordner, der die Envelope.Config-Datei enthält, z.B. C:\Program Files\Altova\MapForce2012\MapForceEDI\X12\ und bearbeiten Sie die Datei.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="...">
3   <Meta>
4     <Release>6020</Release>
5     <Agency>X12</Agency>
6   </Meta>
7   <Format standard="X12"/>
8   <Include href="X12.Segment"/>
9   <Include href="X12.Codelist"/>
10  <Include collection="EDI.Collection"/>
11  <Group name="Envelope">
12    <Group name="Interchange" maxOccurs="unbounded">
13      <Segment ref="ISA" minOccurs="0"/>
14      <Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>
15    <Group name="Group" maxOccurs="unbounded">
```

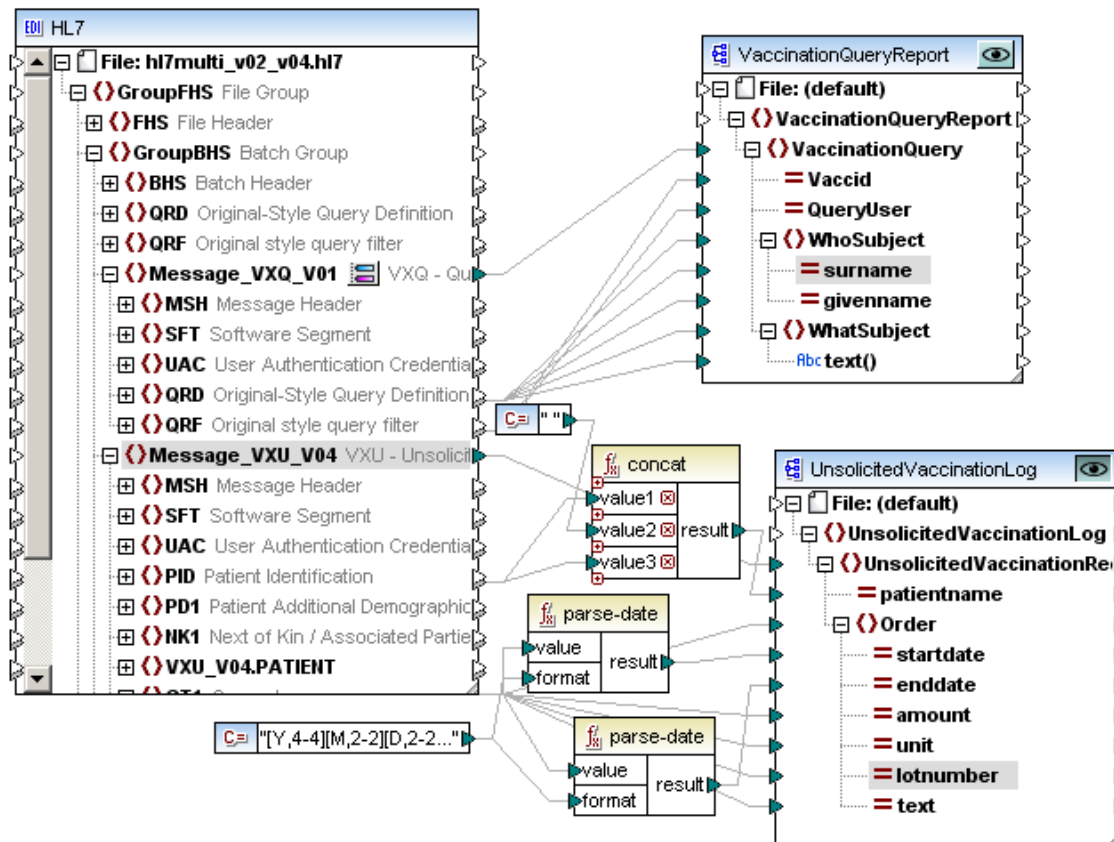
2. Fügen Sie eine Referenz auf das TA1-Segment hinzu, indem Sie zwischen die Segmente "ISA" und "Group" `<Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>` einfügen. Speichern Sie die Datei anschließend.
3. Wenn Sie die Datei X12_To_XML_Order.mfd aus dem Ordner MapForceExamples öffnen, sehen Sie, dass das Segment sowohl zur Bestellkomponente 850 als auch zur Functional Acknowledgment-Komponente 997 hinzugefügt wurde. Sie können nun auswählen, welche TA1-Einträge Sie auf die Zielkomponente mappen möchten.



20.7 Mehrere EDI-Nachrichten pro Komponente

MapForce kann nun mehrere Nachrichtentypen (einer einzigen EDI-Standardversion) pro EDI-Komponente verarbeiten. Auf diese Art können sie folgende Dateien verarbeiten:

- mehrere Dateien (mittels Platzhalterzeichen), die unterschiedliche Nachrichten als Input-Instanzen enthalten
- unterschiedliche Nachrichtentypen innerhalb derselben Datei, d.h. Datenaustausch



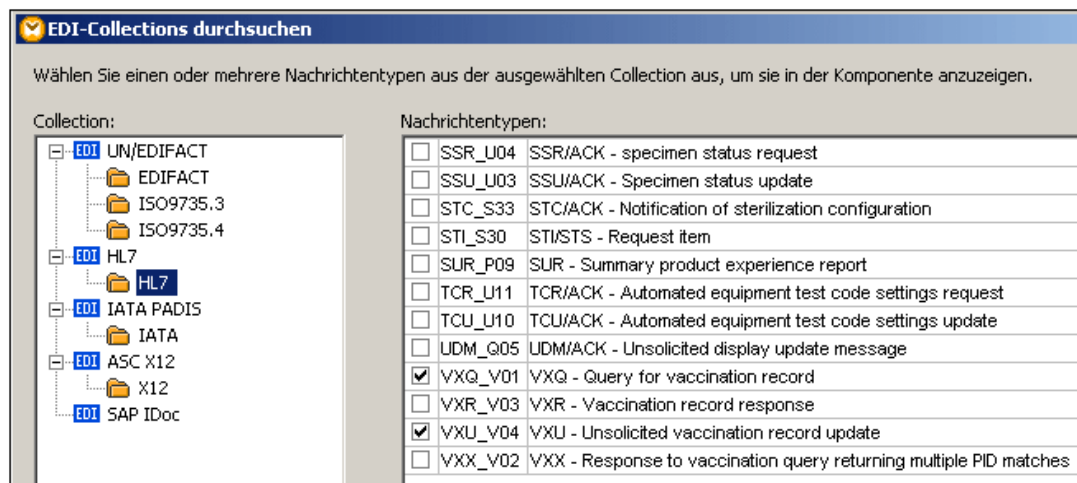
Bitte beachten Sie:

Mit der Schaltfläche "Wählen Sie EDI-Nachrichtentypen aus"  in der EDI-Komponente wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet, in dem Sie mehrere Nachrichten pro Austausch (pro EDI-Komponente) auswählen können.

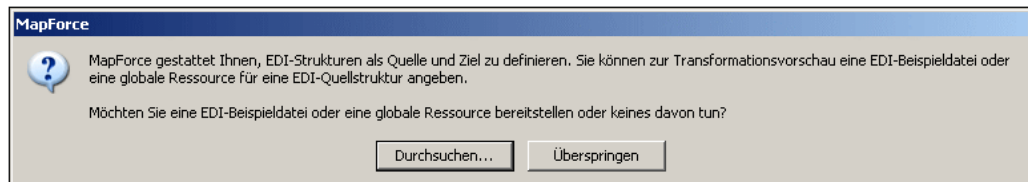
Das oben gezeigte Beispiel finden Sie im [.../MapForceExamples](#) unter dem Namen **HL7_MultiMessageType.mfd**. Im folgenden Abschnitt werden die Dateien aus diesem Ordner verwendet.

So erstellen Sie eine EDI-Komponente, die mehrere Nachrichtentypen enthält:

1. Erstellen Sie ein neues Mapping und stellen Sie sicher, dass entweder Java, C#, C++ oder BUILTIN aktiv ist. Beim Mappen von EDI-Dateien kann kein XSLT 1.0 / 2.0- oder XQuery-Code generiert werden.
2. Wählen Sie die Menüoption **Einfügen | EDI** oder klicken Sie auf das EDI-Symbol und anschließend auf den HL7-Ordner unter dem Eintrag HL7.



3. Aktivieren Sie die Nachrichtentypen, die in der Komponente vorkommen sollen, z.B. VXQ_V01 und VXU_V04 und klicken Sie anschließend auf OK. Sie werden nun aufgefordert, eine EDI-Beispieldatei anzugeben.



4. Klicken Sie auf die Schaltfläche "Durchsuchen" und wählen Sie die EDI-Datei, die die Mapping-Daten liefern soll, z.B. hl7multi_v02_v04.hl7 und klicken Sie auf die Schaltfläche "Öffnen". Daraufhin wird das Dialogfeld "Komponenteneinstellungen" geöffnet, in dem Sie die EDI- oder Kodierungseinstellungen ändern können.
5. Klicken Sie auf OK, um die Standardeinstellungen zu übernehmen und fügen Sie die EDI-Komponente ein.



Die beiden im Dialogfeld "EDI-Collections durchsuchen" ausgewählten Nachrichten werden nun unterhalb des Gruppeneintrags "GroupBHS" als separate Nachrichtenelemente angezeigt, z.B. Message_VXQ_V01 and Message_VXQ_V04

So ändern Sie die Anzahl der Nachrichten in einer Komponente:

1. Klicken Sie neben der ersten Nachricht in der Komponente auf die Schaltfläche "Wählen Sie EDI-Nachrichtentypen aus" . Daraufhin wird das Dialogfeld "EDI-Collections durchsuchen" ausgewählt.
2. Aktivieren (bzw. deaktivieren) Sie die Kontrollkästchen für die Nachrichten, um die Anzahl der Nachrichten zu ändern und klicken Sie anschließend zur Bestätigung auf OK.

EDI-Collections durchsuchen

Wählen Sie einen oder mehrere Nachrichtentypen aus der ausgewählten Collection aus, um sie in der Komponente anzuzeigen.

Collection:	Nachrichtentypen:																																				
[-] EDI UN/EDIFACT - EDIFACT - ISO9735.3 - ISO9735.4 [-] EDI HL7 - HL7 [-] EDI IATA PADIS - IATA [-] EDI ASC X12 - X12 [-] EDI SAP IDoc	<table border="1"> <tbody> <tr><td>☐</td><td>SSR_U04</td><td>SSR/ACK - specimen status request</td></tr> <tr><td>☐</td><td>SSU_U03</td><td>SSU/ACK - Specimen status update</td></tr> <tr><td>☐</td><td>STC_S33</td><td>STC/ACK - Notification of sterilization configuration</td></tr> <tr><td>☐</td><td>STI_S30</td><td>STI/STS - Request item</td></tr> <tr><td>☐</td><td>SUR_P09</td><td>SUR - Summary product experience report</td></tr> <tr><td>☐</td><td>TCR_U11</td><td>TCR/ACK - Automated equipment test code settings request</td></tr> <tr><td>☐</td><td>TCU_U10</td><td>TCU/ACK - Automated equipment test code settings update</td></tr> <tr><td>☐</td><td>UDM_Q05</td><td>UDM/ACK - Unsolicited display update message</td></tr> <tr><td>☒</td><td>VXQ_V01</td><td>VXQ - Query for vaccination record</td></tr> <tr><td>☐</td><td>VXR_V03</td><td>VXR - Vaccination record response</td></tr> <tr><td>☒</td><td>VXU_V04</td><td>VXU - Unsolicited vaccination record update</td></tr> <tr><td>☐</td><td>VXX_V02</td><td>VXX - Response to vaccination query returning multiple PID matches</td></tr> </tbody> </table>	☐	SSR_U04	SSR/ACK - specimen status request	☐	SSU_U03	SSU/ACK - Specimen status update	☐	STC_S33	STC/ACK - Notification of sterilization configuration	☐	STI_S30	STI/STS - Request item	☐	SUR_P09	SUR - Summary product experience report	☐	TCR_U11	TCR/ACK - Automated equipment test code settings request	☐	TCU_U10	TCU/ACK - Automated equipment test code settings update	☐	UDM_Q05	UDM/ACK - Unsolicited display update message	☒	VXQ_V01	VXQ - Query for vaccination record	☐	VXR_V03	VXR - Vaccination record response	☒	VXU_V04	VXU - Unsolicited vaccination record update	☐	VXX_V02	VXX - Response to vaccination query returning multiple PID matches
☐	SSR_U04	SSR/ACK - specimen status request																																			
☐	SSU_U03	SSU/ACK - Specimen status update																																			
☐	STC_S33	STC/ACK - Notification of sterilization configuration																																			
☐	STI_S30	STI/STS - Request item																																			
☐	SUR_P09	SUR - Summary product experience report																																			
☐	TCR_U11	TCR/ACK - Automated equipment test code settings request																																			
☐	TCU_U10	TCU/ACK - Automated equipment test code settings update																																			
☐	UDM_Q05	UDM/ACK - Unsolicited display update message																																			
☒	VXQ_V01	VXQ - Query for vaccination record																																			
☐	VXR_V03	VXR - Vaccination record response																																			
☒	VXU_V04	VXU - Unsolicited vaccination record update																																			
☐	VXX_V02	VXX - Response to vaccination query returning multiple PID matches																																			

20.8 UN/EDIFACT und ANSI X12 als Zielkomponenten

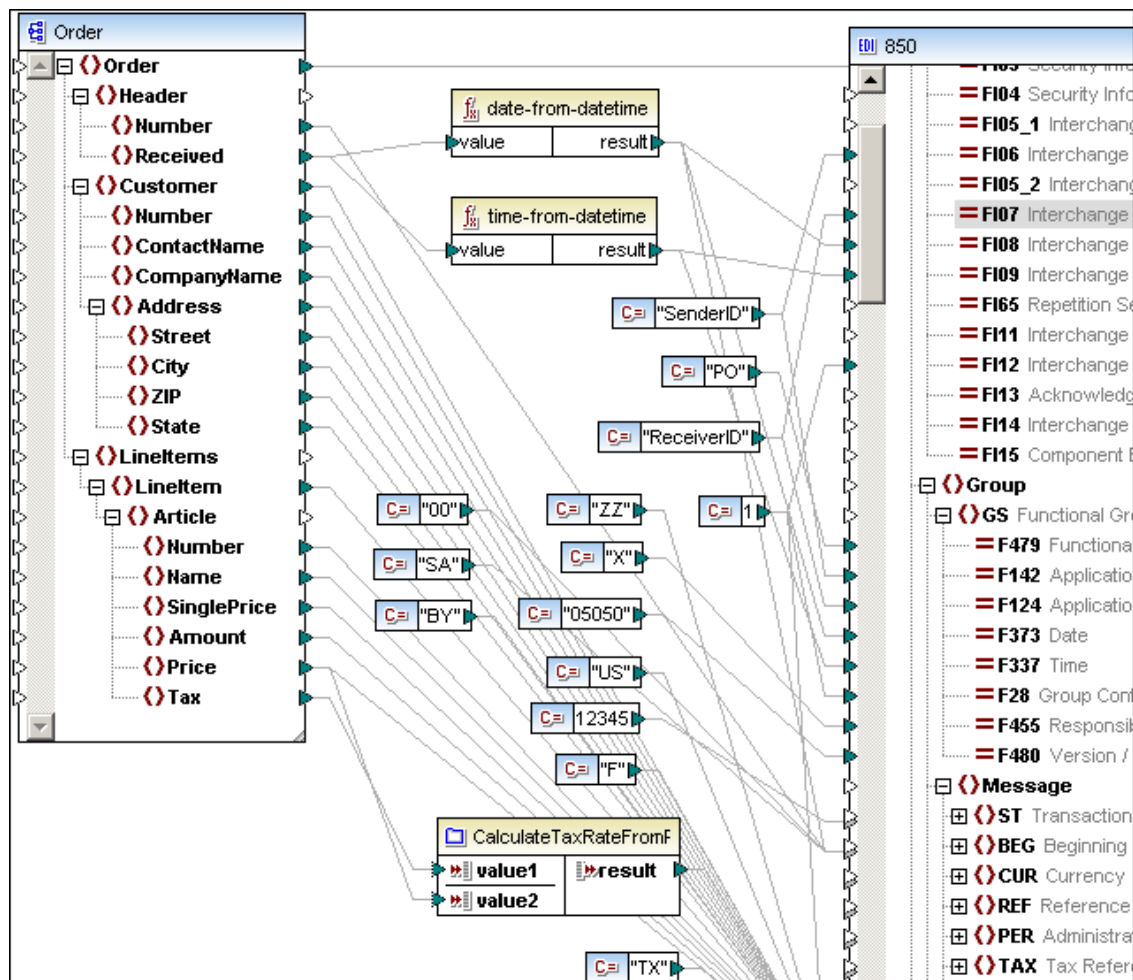
In der Beispieldatei **XML_To_X12_Order.mfd** im Ordner **...MapForceExamples** wird eine XML-Datei auf eine ANSI X12 EDI-Datei gemappt. Die erzeugte EDI-Ausgabedatei kann gegen die X12-Spezifikationen validiert werden, indem Sie auf das Symbol "Ausgabedatei validieren" klicken.

Der folgende Abschnitt enthält eine Beschreibung:

- der gemeinsamen Einstellungen für die beiden EDI-Formate
- der Validierungsregeln und der Einstellungen zur automatischen Datenkomplettierung für die spezifischen EDI-Formate

Bitte beachten Sie:

Wenn die Reihenfolge der (Daten)elemente in der Quellkomponente beibehalten werden soll, stellen Sie bitte sicher, dass die Konnektoren mit der Option "Quellorientiertes Mapping (Mixed Content)" erstellt werden. Nähere Informationen dazu siehe Abschnitt [Quellorientiertes und Mixed Content Mapping](#).



Automatische Datenkomplettierung

Bei der automatischen Datenkomplettierung werden bestehende Daten im Allgemeinen nicht geändert. Es werden jedoch bestimmte obligatorische Nodes erstellt und es werden nach dem Mapping ggf. Daten eingefügt, um ein gültiges Dokument zu erzeugen.

Bitte beachten Sie, dass Felder, die in den Abschnitten zur automatischen Datenkomplettierung nicht aufgelistet sind, NICHT eingefügt oder erstellt werden. Die korrekten Werte können nicht automatisch ermittelt werden.

Allgemeine Einstellungen:

Sie können einige allgemeine Einstellungen festlegen, die für alle EDI-Dokumente gelten:

- Autokomplettierung von fehlenden Feldern:
Soll die Automatische Ergänzung von Daten aktiviert werden oder nicht?
Standardeinstellung: **true**
- Nach jedem Segment neue Zeile beginnen:
Soll zur besseren Lesbarkeit eine neue Zeile an jedes Segment angehängt werden?
Der EDI-Standard ignoriert diese Zeilenendezeichen, falls sie in einer Message vorhanden sind. Standardeinstellung: **true**
- Datenelement-Trennzeichen: siehe EDIFACT/X12-Spezifikation.
Standardeinstellung: **+**
- Composite-Trennzeichen: siehe EDIFACT/X12-Spezifikation.
Standardeinstellung :
- Segment-Abschlusszeichen: siehe EDIFACT/X12-Spezifikation.
Standardeinstellung '
- Dezimaltrennzeichen: siehe EDIFACT/X12-Spezifikation.
Standardeinstellung .
- Release-Zeichen: siehe EDIFACT/X12-Spezifikation.
Standardeinstellung ?
- HL7 unterstützt außerdem einen zusätzlichen Trennzeichen für untergeordnete Felder, wobei das Standardzeichen **&** ist. Dieses Trennzeichen wird im Dialogfeld "Komponenteneinstellungen" als Unterkomponenten-Trennzeichen bezeichnet.

Klicken Sie mit der rechten Maustaste auf eine EDI-Komponente und wählen Sie den Befehl **Eigenschaften**, um das Dialogfeld "EDI-Komponenteneinstellungen" zu öffnen. Die UN/EDIFACT-Einstellungen werden für beide EDI-Komponenten als Standardeinstellung verwendet.

Die in dieses Dialogfeld eingegebenen EDI-Trennzeichen wirken sich immer beim **Schreiben** der EDI-Dateien aus. Beim **Lesen** in EDI-Dateien, wirken sich die Trennzeichen nur aus, wenn in der Input-Datei keine eigenen Trennzeichen definiert/enthalten sind, z.B. in EDIFACT-Dateien ohne das UNA "service string advice" Segment.

Wenn eine EDI-Input-Komponente/Datei Trennzeichendefinitionen enthält, z.B. eine X12-Datei mit einem ISA-Segment, so setzen die vorhandenen Trennzeichen alle im Dialogfeld "Komponenteneinstellungen" für diese Datei definierten Trennzeichen außer Kraft.

Wenn Sie auf die Schaltfläche "Erweitert..." klicken, wird das erweiterte Dialogfeld für die jeweiligen EDI-Einstellungen geöffnet.

Definieren nicht druckbarer Zeichen:

Sie können nicht druckbare Zeichen als Trennzeichen verwenden. Geben Sie dazu "x" gefolgt vom hexadezimalen ASCII-Zeichencode in eines der Auswahllistenfelder ein, z.B. "x1e" für das RS Control-Zeichen (ASCII-Datensatztrennzeichen, Dezimalcode 30).

Mappen von Datums- und Uhrzeit-Datentypen

Vor der Version MapForce 2012 R3 waren Datum und Uhrzeit in EDIFACT- und X12-Komponenten vom Typ "string". Ab dieser Release können Datum und Uhrzeit direkt von/auf **xsd:time** bzw. **xsd:date** gemappt werden (und auch von SQL date/time).

Bei benutzerdefinierten Funktionen vor der Version 2006R3, bei denen `xsd:date` oder `xsd:time` in ein EDI-Format konvertiert werden, wird eine Fehlermeldung generiert. Sie können daher in dieser Form nicht verwendet werden. Verwenden Sie bitte den Schema-Datentyp **xsd:dateTime**, der mit Hilfe der integrierten Funktionen **date-from-datetime** bzw. **time-from-datetime** aus der datetime-Bibliothek gemappt werden kann.

Wenn Datumsangaben innerhalb von benutzerdefinierten Funktionen gemappt werden müssen, stellen Sie bitte sicher, dass diese dem ISO 8601 date/time-Format entsprechen, d.h. JJJJ-MM-TT.

Validierung

Die EDI-Validierung wird sowohl auf dem Register "Ausgabe" (BUILTIN) als auch im generierten Code unterstützt. Wählen Sie den Menübefehl "**Ausgabe | Ausgabedatei validieren**" um die EDI-Ausgabedatei auf dem Register "Ausgabe" zu validieren.

Bei Ausführung des generierten Programmcodes lösen Parsing-Fehler ein Ausnahmeereignis aus, wodurch das Mapping gestoppt wird und aufgrund der Validierungsfehler wird eine entsprechende Meldung in der Standardausgabe angezeigt.

Dies kann durch Bearbeitung der SPL-Vorlagen (des generierten Codes) oder durch Ändern der Werte für `ErrorMask` und `WarningMask` behoben werden. `ErrorMask` definiert, welche Arten von Fehlern ein Ausnahmeereignis auslösen. `WarningMask` definiert, welche Arten von Fehlern Warnmeldungen in der Standardausgabe erzeugen. Nähere Informationen dazu finden Sie in den SPL-Vorlagen in den Kommentaren zu "ErrorMask".

20.8.1 Validierung des UN/EDIFACT-Zieldokuments

Die folgenden Punkte werden bei der Validierung eines UN/EDIFACT-Dokuments überprüft.

- ob ein UNB- und ein UNZ-Segment existieren.
- ob UNB/S004 eine gültige Datums-/Uhrzeit-Angabe enthält.
- ob UNB/0020 und UNZ/0020 denselben Wert enthalten.
- ob UNZ/0036 die richtige Zahl, die der Anzahl der Funktionsgruppen, falls vorhanden, bzw. der Anzahl der Messages entspricht. Wenn Funktionsgruppen vorhanden sind, sollte dies die Anzahl der Funktionsgruppen sein, andernfalls sollte dies die Anzahl der im Datenaustausch (Interchange) enthaltenen Messages sein.

Jede **Funktionsgruppe** wird auf folgende Punkte überprüft:

- ob sie ein übereinstimmendes UNG- und UNE-Paar enthält.
- ob UNG/S004 eine gültige Datums-/Uhrzeit-Angabe enthält.
- ob UNE/0060 die richtige Zahl enthält, die der Anzahl der Messages in der Funktionsgruppe entsprechen muss.

Jede **Message** wird auf folgende Punkte überprüft:

- ob sie ein übereinstimmendes UNH- und UNT-Paar enthält.
- ob UNH/S009/0052 denselben Wert enthält wie UNG/S008/0052 der einschließenden Funktionsgruppe.
- ob UNH/0062 und UNT/0062 denselben Wert enthalten.
- ob UNH/S009/0065 den korrekten Message Type Specifier enthält.
- ob UNT/0074 die korrekte Anzahl der in der Message enthaltenen Segmente enthält.

Die **automatische Datenkomplettierung** für EDIFACT stellt sicher:

- dass ein UNB- und ein UNZ-Segment existieren.
- dass, wenn entweder UNG oder UNE vorhanden ist, die andere ID ebenfalls vorhanden ist.
- dass ein UNH- und ein UNT-Segment vorhanden sind.
- dass UNB/S001 existiert. Wenn es keine Daten enthält, werden Syntax Level und Syntax Versionsnummer aus den benutzerdefinierten Einstellungen verwendet. Siehe **Erweitert | Syntax Version Number**.
- dass UNB/S002 und UNB/S003 vorhanden sind.
- dass UNB/S004 vorhanden ist. Wenn es keine Daten enthält, wird das aktuelle Datum/die aktuelle Uhrzeit im EDI-Format eingefügt.
- dass UNZ/0036 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Funktionsgruppen und Messages berechnet und eingefügt.
- dass UNZ/0020 vorhanden ist. Wenn es keine Daten enthält, wird der Wert aus UNB/0020 kopiert.

Bitte beachten Sie:

Felder, die in diesem Abschnitt (Automatische Datenkomplettierung) nicht erwähnt sind, werden NICHT eingefügt oder erstellt. Die korrekten Werte können nicht automatisch ermittelt werden.

Bei einem Parent-(Ziel)-Element A (in der EDI-Zielkomponente) mit den Child-Elementen x, y und z - wobei y obligatorisch ist - wird das Parent-Element A in der Ausgabedatei nur dann erstellt, wenn das obligatorische Child-Element "y" in der Zielkomponente gemappt wurde!

Bei der Überprüfung der Funktionsgruppen wird sichergestellt:

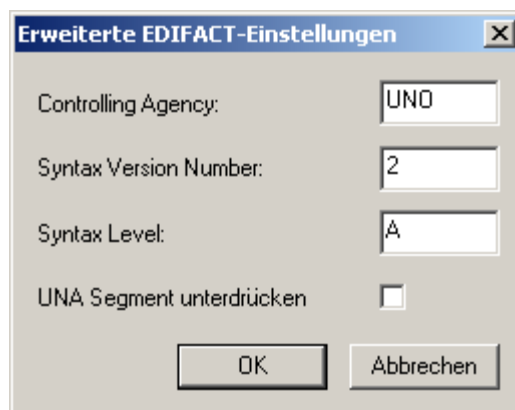
- dass UNG/0038 vorhanden ist. Wenn es keine Daten enthält, wird der Name der Message eingefügt.
- dass UNG/S006 und UNG/S007 vorhanden sind.
- dass UNG/S004 vorhanden ist. Wenn es keine Daten enthält, wird das aktuelle Datum/die aktuelle Uhrzeit im EDI-Format eingefügt.
- dass UNG/0051 vorhanden ist. Wenn es keine Daten enthält, werden aus den benutzerdefinierten Einstellungen die ersten beiden Zeichen oder Controlling Agency eingefügt. Siehe **Einstellungen | Controlling agency**.
- dass UNE/0060 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Messages in der Gruppe berechnet und eingefügt.
- dass UNE/0048 vorhanden ist. Wenn es keinen Wert enthält, wird der Wert aus UNG/0048 kopiert.

Bei der Überprüfung der Messages wird sichergestellt:

- dass UNH/S009/0065 vorhanden ist. Wenn es keine Daten enthält, wird der Name der Message eingefügt.
- dass UNH/S009/0052 und UNH/S009/0054 vorhanden sind.
- dass UNH/S009/0051 vorhanden ist. Wenn es keine Daten enthält, werden aus den benutzerdefinierten Einstellungen die ersten beiden Zeichen oder Controlling Agency eingefügt. Siehe **Einstellungen | Controlling agency**.
- dass UNT/0074 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Segmente in der Message berechnet und eingefügt.
- dass UNT/0062 vorhanden ist. Wenn es keinen Wert enthält, wird der Wert aus UNH/0062 kopiert.
- Dass UNH/0062 vorhanden ist. Wenn es keinen Wert enthält, wird der Wert aus UNT/0062 kopiert.
(Wenn nur die Nummer des Trailer Segments gemappt wird, erhält das entsprechende Feld im Header-Segment denselben Wert)

Einstellungen

Wenn Sie im Dialogfeld "Komponenteneinstellungen" auf die Schaltfläche "Erweitert..." klicken, wird das Dialogfeld für die erweiterten Einstellungen geöffnet. Nähere Informationen dazu finden Sie in den UN/EDIFACT-Spezifikationen.



The screenshot shows a Windows-style dialog box titled "Erweiterte EDIFACT-Einstellungen". It has a close button (X) in the top right corner. Inside the dialog, there are four configuration items, each with a label and a text input field or checkbox:

- Controlling Agency: UNO
- Syntax Version Number: 2
- Syntax Level: A
- UNA Segment unterdrücken: ☐

At the bottom of the dialog, there are two buttons: "OK" and "Abbrechen".

- Controlling Agency: Standardeinstellung **UNO**
- Syntax Version Number: Standardeinstellung **4**
- Syntax Level: Standardeinstellung **A**
- UNA Segment unterdrücken: Standardeinstellung **deaktiviert**.

20.8.2 Validierung des ANSI X12-Zieldokuments

Bei der Validierung eines ANSI X12-Dokuments werden die folgenden Punkte überprüft:

- ob ein ISA- und ein IEA-Segment vorhanden sind
- ob ISA/I01 einen zulässigen Authorization Information Qualifier enthält.
- ob ISA/I03 einen zulässigen Security Information Qualifier enthält.
- ob die beiden ISA/I05 Segmente zulässige Interchange ID Qualifier enthält.
- ob ISA/I08 einen wohlgeformten Datumswert enthält.
- ob ISA/I09 einen wohlgeformten Uhrzeitwert enthält.
- ob ISA/I13 einen zulässigen Booleschen Wert enthält.
- ob ISA/I14 einen zulässigen Interchange Usage Indicator enthält.
- ob ISA/I12 und IEA/I12 denselben Wert enthalten.
- ob IEA/I16 die richtige Anzahl von Funktionsgruppen im Interchange enthält.

Jede **Funktionsgruppe** wird auf folgende Punkte überprüft:

- ob sie ein übereinstimmendes GS- und GE-Paar enthält.
- ob GS/373 einen wohlgeformten Datumswert enthält.
- ob GS/337 einen wohlgeformten Uhrzeitwert enthält.
- ob GS/28 und GE/28 denselben Wert enthalten.
- ob GE/97 die richtige Anzahl von Transaction Sets in der Funktionsgruppe enthält.

Jede **Message** wird auf folgende Punkte überprüft:

- ob es ein übereinstimmendes ST- und SE-Paar gibt.
- ob ST/143 den richtigen Message Identifier enthält.
- ob ST/329 und SE/329 denselben Wert enthalten.
- ob SE/96 die richtige Anzahl von Segmenten in der Message enthält.

Die automatische Datenkomplettierung für EDI/X12 stellt sicher:

- dass auf dem Interchange Level ein ISA- und IEA-Paar vorhanden sind.
- dass bei Vorhandensein von entweder GS oder GE auch die andere ID vorhanden ist.
- dass es auf dem Message Level mindestens ein ST/SE-Paar gibt.
- dass ISA/I01 und ISA/I03 vorhanden sind. Wenn sie keine Daten enthalten, wird 00 eingesetzt.
- dass ISA/I02 und ISA/I04 vorhanden sind. Wenn sie keine Daten enthalten, werden zehn Leerstellen eingesetzt.
- dass beide ISA/I05 Segmente vorhanden sind. Wenn sie keine Daten enthalten, wird ZZ eingesetzt.
- dass ISA/I08 vorhanden ist. Wenn es keine Daten enthält, wird das aktuelle Datum im EDI-Format eingesetzt.
- dass ISA/I09 vorhanden ist. Wenn es keine Daten enthält, wird die aktuelle Uhrzeit im

EDI-Format eingesetzt.

- dass ISA/I65 vorhanden ist. Wenn es keine Daten enthält, wird das Wiederholungstrennzeichen eingesetzt.
- dass ISA/I11 vorhanden ist. Wenn es keine Daten enthält, wird die Interchange Control Version Number aus den benutzerdefinierten Einstellungen eingesetzt. Siehe **Einstellungen | Interchange Control Version-Number**.
- dass ISA/I12 vorhanden ist.
- dass ISA/I13 vorhanden ist. Wenn es keine Daten enthält, wird die Request Acknowledgement-Einstellung verwendet Siehe **Einstellungen | Request Acknowledgement**.
- dass ISA/I14 vorhanden ist. Wenn es keine Daten enthält, wird P eingesetzt.
- dass ISA/I15 vorhanden ist. Wenn es keine Daten enthält, wird das Composite-Trennzeichen aus den benutzerdefinierten Einstellungen eingesetzt. Siehe **Einstellungen | Composite-Trennzeichen**.
- dass IEA/I16 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Funktionsgruppen im Interchange berechnet und eingesetzt.
- dass IEA/I12 vorhanden ist. Wenn es keine Daten enthält, wird der Wert aus ISA/I12 kopiert.

Bitte beachten Sie:

Felder, die in diesem Abschnitt (Automatische Datenkomplettierung) nicht erwähnt sind, werden NICHT eingefügt oder erstellt. Die korrekten Werte können nicht automatisch ermittelt werden.

Bei einem Parent-(Ziel)-Element A (in der EDI-Zielkomponente) mit den Child-Elementen x, y und z - wobei y obligatorisch ist - wird das Parent-Element A in der Ausgabedatei nur dann erstellt, wenn das obligatorische Child-Element "y" in der Zielkomponente gemappt wurde!

Potentiell existierende Funktionsgruppen werden auf folgenden Punkte überprüft:

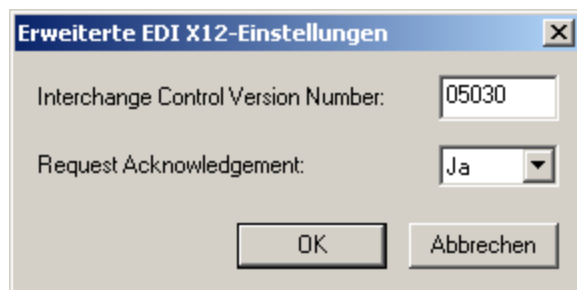
- dass GS/373 vorhanden ist. Wenn es keine Daten enthält, wird das aktuelle Datum im EDI-Format eingefügt.
- dass GS/337 vorhanden ist. Wenn es keine Daten enthält, wird die aktuelle Uhrzeit im EDI-Format eingesetzt.
- dass GE/97 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Messages in der Funktionsgruppe berechnet und eingesetzt.
- dass GE/28 vorhanden ist. Wenn es keine Daten enthält, wird der Wert aus GS/28 kopiert.

Bei der Überprüfung von Messages wird sichergestellt:

- dass ST/143 vorhanden ist. Wenn es keine Daten enthält, wird der Name der Message eingefügt.
- dass SE/96 vorhanden ist. Wenn es keine Daten enthält, wird die Anzahl der Segmente in der Message berechnet und eingesetzt.
- dass ST/329 und SE/329 vorhanden sind. Wenn SE/329 keine Daten enthält, wird der Wert aus ST/329 kopiert.

Einstellungen

Wenn Sie im Dialogfeld "Komponenteneinstellungen" auf die Schaltfläche "Erweitert..." klicken, wird das Dialogfeld für erweiterte Einstellungen geöffnet. Nähere Informationen dazu finden Sie in den EDI/X12-Spezifikationen.



- Interchange Control Version Number: Standardeinstellung **06020**
- Request Acknowledgement: Standardeinstellung **ja**

20.9 Anpassen einer EDIFACT Message

MapForce gestattet Ihnen, EDIFACT Messages an unterschiedliche, nicht-standardkonforme oder geänderte EDIFACT-Formate anzupassen.

In diesem Beispiel wird die Datei **Orders-Custom-EDI.mfd** aus dem Ordner **...\MapForceExamples\Tutorial** verwendet. Beachten Sie bitte, dass der Ordner **..\Tutorial** auch die ZIP-Datei **EDIFACT.Nanonull.zip** enthält. Die ZIP-Datei enthält die Datei **result** in der das Ergebnis der Anpassung gespeichert ist.

In der oben erwähnten EDIFACT-Datei im Ordner **..\Tutorial** wurde zum CTA-Segment ein neues Komponentenelement hinzugefügt:

- Zeile 9 enthält einen Eintrag **Mr** und
- Zeile 11 einen Eintrag **Mrs**.

1	UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
	ADDR+970101:1230+000000000000001++ORDERS++++1'
2	UNH+0001+ORDERS:S:93A:UN'
3	BGM+221+ABCL123456XYZ+9'
4	DTM+4:200404301742PDT:303'
5	FTX+PUR+3++Pizza purchase order'
6	RFF+CT:123-456'
7	RFF+CR:1122'
8	NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+CA+34424+US'
9	CTA+SR+:Ted Little:Mr'
10	NAD+BY+123::92++Nanonull, Inc.+119 Oakstreet Suite 4876+Vereno+CA+29213+US'
11	CTA+PD+:Michelle Butler:Mrs'
12	NAD+ST+123::92++Nanonull, Inc.+119 Oakstreet Suite 4876+Vereno+CA+29213+US'
13	TAX+9+++++1-12345-6789-0'
14	CUX+2:USD:9'
15	PCD+12:2'
16	TDT+20++++::24h Pizza Fast Carrier'
17	LOC+16+Nanonull Main Entrance'

Das CTA (Contact Information)-Segment der ORDERS Message muss erweitert werden, damit die neuen Daten gemappt werden können.

20.9.1 EDIFACT: Anpassung

In den folgenden Abschnitten wird beschrieben, wie man die Konfigurationsdateien anpasst, um die geänderte EDIFACT Message auf das Schema **ORDER-EDI.xsd** mappen zu können. Es wird davon ausgegangen, dass die mitgelieferte ZIP-Datei **nicht** verwendet wird.

So bereiten Sie das Beispiel für die Anpassung der EDIFACT Message vor:

1. Erstellen Sie unter **Program Files...\MapForceEDI\EDIFACT.Nanonull** einen Ordner EDIFACT.Nanonull
2. Kopieren Sie die folgenden Dateien aus dem Ordner ...**MapForceEDI\EDIFACT** in den Ordner EDIFACT.Nanonull:

Admin.Segment
EDI.Collection
EDSD.Segment
Envelope.Config
ORDERS.Config
UNCL.Codelist

3. Ändern Sie die Attribute der Dateien in Lese/Schreibzugriff, um sie editierbar zu machen.

Konfigurieren der Datei EDI.Collection:

1. Öffnen Sie die Datei "EDI.Collection" in XMLSpy oder in einem Editor Ihrer Wahl.
2. Entfernen Sie alle "Message"-Elemente mit Ausnahme der ORDERS Message und stellen Sie sicher, dass die <Messages>-Tags erhalten bleiben!

```

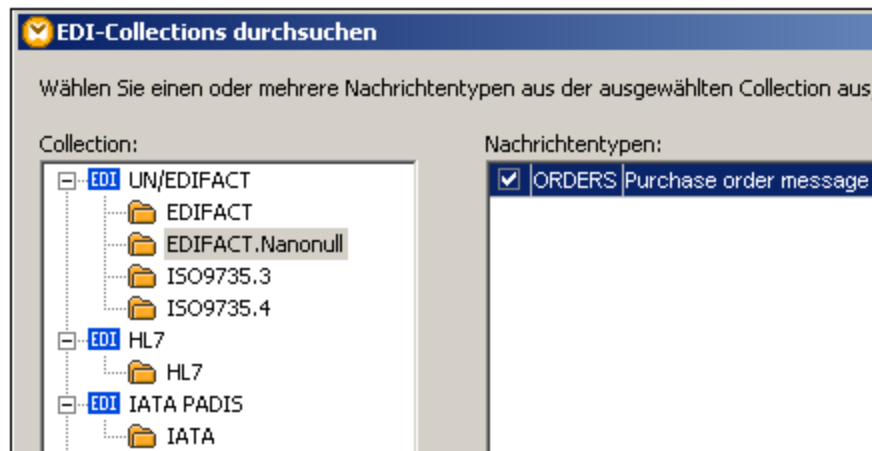
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Messages Version="3">
3    <Meta>
4      <Version>D</Version>
5      <Release>10A</Release>
6      <Agency>UN</Agency>
7    </Meta>
8    <Root File="Envelope.Config"/>
9    <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase order message"/>
10 </Messages>

```

3. Speichern Sie die Datei.

So zeigen Sie den Inhalt der Collection-Datei an:

- Starten Sie MapForce, wählen Sie **Einfügen | EDI** oder klicken Sie auf das Symbol "EDI-Datei einfügen". Daraufhin wird das Dialogfeld "EDI-Collection durchsuchen" geöffnet, das ein neuer Ordner mit dem Namen "EDIFACT.Nanonull" enthält. Sie sehen nur einen Eintrag; die erste Spalte enthält den Message-Typ "ORDERS", die zweite den beschreibenden Text zur Message.



Bitte beachten Sie:

MapForce durchsucht alle Unterordner auf gleicher Ebene im Verzeichnis "..\MapforceEDI" und sucht nach einer Datei namens "EDI.Collection". Der Ordnername mit einer EDI Collection wird als Ordner im Dialogfeld angezeigt. Das Listenfeld enthält den aktuellen Inhalt der Collection-Datei, die in unserem Beispiel nur den Message-Eintrag ORDERS enthält.

Ziel dieses Abschnitts ist es, das CTA (Contact Information) Segment durch Hinzufügen des Felds X1000 neu zu definieren, damit es für einzelne Messages verfügbar ist. CTA besteht aus einem Feld (F3139) und einem Composite (C056).

Es gibt verschiedene Methoden, um die Anpassung vorzunehmen:

- **Global** durch Anpassen der Datei **EDSD.segment**. Alle Segmente in allen Messages, in denen das Composite C056 verwendet wird, enthalten daraufhin das neue Element bzw. eine Referenz darauf.
- **Inline** durch Anpassen der Datei **ORDERS.Config**. Nur das angepasste Segment (CTA) in der aktuellen Message enthält daraufhin das neue Element.

UNCL.Codelist

In dieser Datei sind die verschiedenen EDIFACT-Codes und die Werte, die sie enthalten können, definiert. Sie wird zur Validierung der Input/Output-Dateien in MapForce verwendet. Falls Ihr Unternehmen einen Sondercode verwendet, der nicht in der EDIFACT-Codeliste ist, fügen Sie ihn hier hinzu.

EDI.Collection

Die Collection-Datei enthält eine Liste aller Messages im aktuellen Verzeichnis. Sie dient dazu eine Liste der verfügbaren Messages abzurufen, wenn eine EDIFACT-Datei in MapForce eingefügt wird. Das folgende Beispiel enthält nur eine Message, nämlich "ORDERS".

Bearbeiten Sie diese Datei, so dass sie nur die Messages enthält, die für Ihre Arbeit benötigt werden.

Orders.Config

Die Konfigurationsdatei für Bestellungs-Message-Dateien. Diese Datei enthält alle Gruppen und Segmentdefinitionen, die in Bestell-Messages verwendet werden. Mit Änderungen an dieser Datei können lokale oder inline Anpassungen definiert werden.

Admin.Segment

In "Admin.segment" werden die für den Message-Austausch erforderlichen administrativen Segmente beschrieben. Diese Segmente werden alle zum Parsen der EDIFACT-Datei verwendet.

EDSD.Segment (Electronic Data Segment Definition)

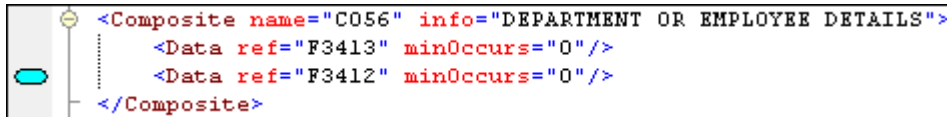
In dieser Datei werden die Segment-, Composite- und Feldnamen der EDIFACT-Dateien beschrieben. Sie dient zum Parsen der EDIFACT-Datei. Änderungen an dieser Datei sind globale Anpassungen und gelten für alle Segmente und Messages.

20.9.2 Globale Anpassung

- Änderungen müssen nur an der Datei **EDSD.Segment** vorgenommen werden, um globalen Zugriff auf das neue Feld X1000 zu haben.
- Alle **Segmente** in allen **Messages**, in denen Composite C056 verwendet wird, werden daraufhin das neue Element bzw. eine Referenz darauf enthalten.

Redefinition des Composite in der Datei EDSD.Segment:

Öffnen Sie die Datei EDSD.Segment in XMLSpy oder dem Editor Ihrer Wahl und navigieren Sie zu **Config | Elements | Composite | C056**.

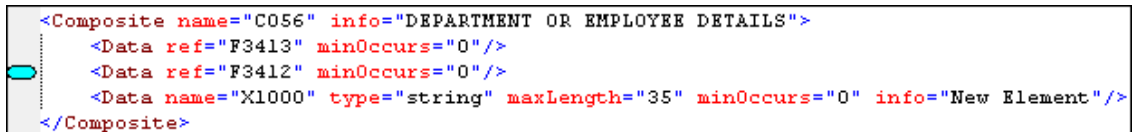


```
<Composite name="C056" info="DEPARTMENT OR EMPLOYEE DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
</Composite>
```

Fügen Sie im Segment C056 unter F3412 die folgende Zeile hinzu:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
```

Die Composite-Definition sieht folgendermaßen aus:



```
<Composite name="C056" info="DEPARTMENT OR EMPLOYEE DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
  <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
</Composite>
```

Bitte beachten Sie:

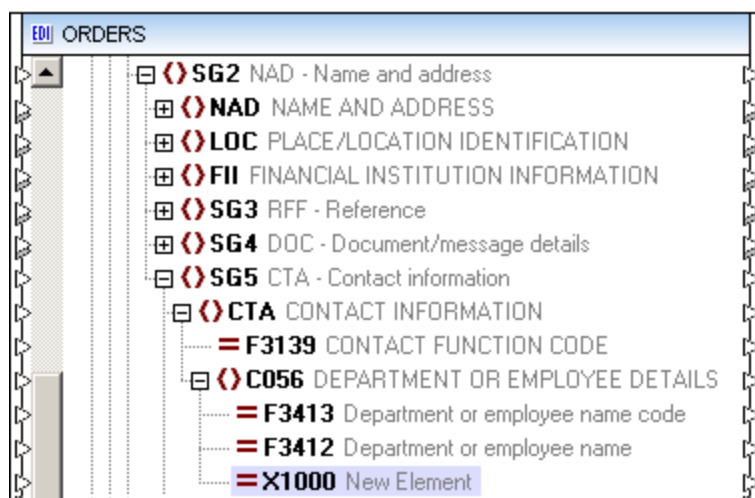
Das neue Feld X1000 wird im Gegensatz zu anderen Feldern, für die das "ref" Attribut verwendet wird, mit Hilfe des "name" Attributs definiert. Die beiden anderen Felder werden am Beginn der EDSD-Segmentdatei außerhalb des Composite-Abschnitts (mit Hilfe des Data name-Elements) definiert und hier nur referenziert. Das neue Feld kann nun von verschiedenen Segmenten oder Composites aus referenziert werden.

Beim Anpassen Ihrer EDI-Dateien können als Werte, die z.B. für das Attribut "type" verwendet werden können, im allgemeinen alle XML-Schematypen verwendet werden, die in den verfügbaren config-Dateien zur Verfügung stehen. Andere XML-Schema simpleTypes können unter Umständen verwendet werden, sind aber nicht immer gültig.

Nicht unterstützte simpleTypes sind: "anyType", "ENTITIES" und "QName".

Vorschau des neuen Felds in MapForce:

1. Wählen Sie die Menüoption **Einfügen | EDI**, oder klicken Sie auf das Symbol . "EDI-Datei einfügen"
2. Klicken Sie auf das Register "EDIFACT.Nanonull" und wählen Sie die ORDERS Message aus.
3. Klicken Sie auf die Schaltfläche "Überspringen", um die Auswahl der EDI-Quelldatei im Moment auszulassen. Die ORDERS-Komponente ist nun im Mapping-Fenster sichtbar.
4. Navigieren Sie zu **Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056**, um das neue X1000 Element zu sehen.



20.9.3 Inline-Anpassung

- Änderungen müssen nur in der Datei **ORDERS.config** an der angegebenen Stelle vorgenommen werden, um inline-Zugriff auf das neue Feld X1000 zu haben.
- Nur das neu definierte CTA-Segment in der **aktuellen** Message enthält/referenziert das neue Element.
- In anderen Worten, das CTA-Segment wird lokal neu definiert und enthält ein neu definiertes Composite C056 mit der lokalen Definition des neuen Felds X1000.

Öffnen Sie die Datei ORDERS.Config in XMLSpy oder dem Editor Ihrer Wahl und navigieren Sie zu **Message | Envelope | Interchange | Group | Message | SG2 | SG5 | CTA** (oder suchen Sie **SG5**).

```

47 <Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
48   <Segment ref="CTA"/>
49   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
50 </Group>

```

Ersetzen Sie die Zeile:

```
<Segment ref="CTA"/>
```

durch die folgenden Zeilen:

```

<Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
  <Data ref="F3139" minOccurs="0"/>
  <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE DETAILS"
">
    <Data ref="F3413" minOccurs="0"/>
    <Data ref="F3412" minOccurs="0"/>
    <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
  </Composite>
</Segment>

```

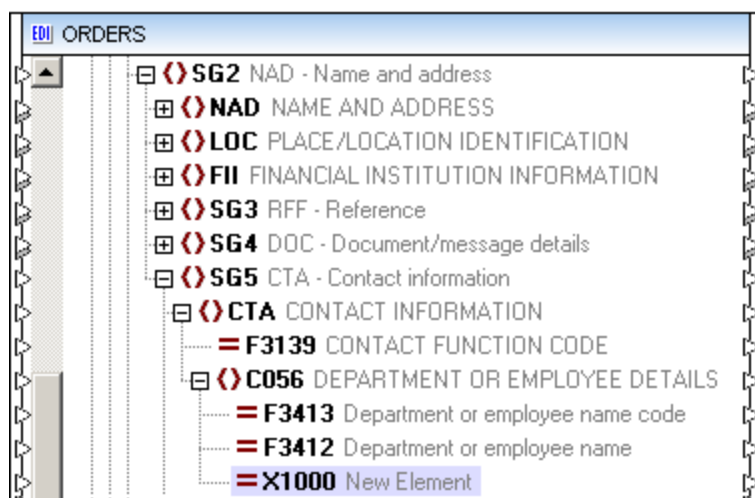
```

47 <Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
48   <Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
49     <Data ref="F3139" minOccurs="0"/>
50     <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE DETAILS">
51       <Data ref="F3413" minOccurs="0"/>
52       <Data ref="F3412" minOccurs="0"/>
53       <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Ele
54     </Composite>
55   </Segment>
56   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
57 </Group>

```

Vorschau des neuen Felds in MapForce:

1. Wählen Sie die Menüoption **Einfügen | EDI**, oder klicken Sie auf das Symbol . "EDI-Datei einfügen".
2. Klicken Sie auf das Register "EDIFACT.Nanonull" und wählen Sie die ORDERS Message aus.
3. Klicken Sie auf die Schaltfläche "Überspringen", um die Auswahl der EDI-Quelldatei im Moment auszulassen. Die ORDERS-Komponente ist nun im Mapping-Fenster sichtbar.
4. Navigieren Sie zu **Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056**, um das neue X1000 Element zu sehen.



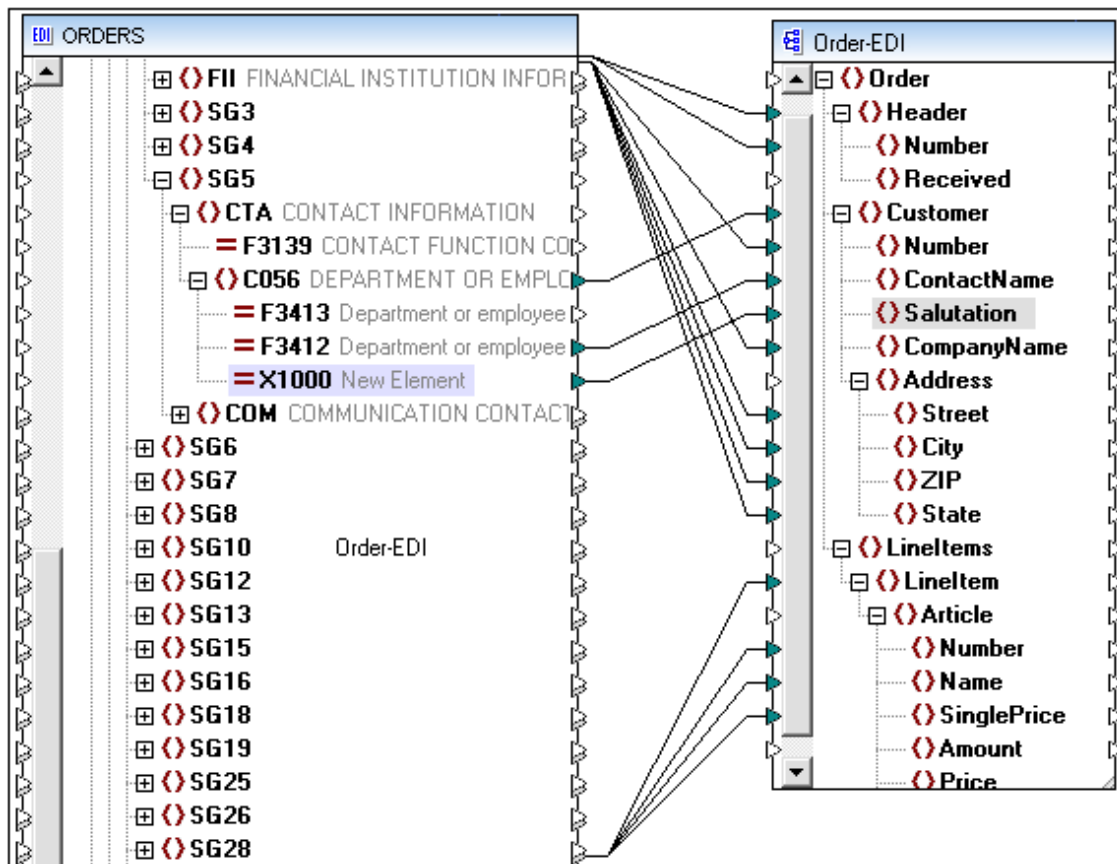
20.9.4 Mapping-Beispiel Orders

Das in den Abbildungen unten gezeigte Mapping Orders-Custom-EDI.mfd steht im Verzeichnis **...MapForceExamples\Tutorial** zur Verfügung.

In dem Beispiel wird die Datei ORDERS-Custom.EDI auf das Schema Order-EDI gemappt. Das Feld X1000, das zu EDI-Struktur hinzugefügt wurde, wurde auf das Datenelement "Salutation" gemappt.

So sehen Sie das Ergebnis der Anpassung:

1. Erstellen Sie im Verzeichnis "**...MapForceEDI**" einen neuen Ordner und geben Sie ihm z.B. den Namen "**EDIFACT.Nanonull**"
2. Entpacken Sie die mitgelieferte Datei EDIFACT.Nanonull.zip (aus dem Ordner **...\Tutorial**) in den neuen Ordner. Die ZIP-Datei enthält die folgenden Dateien:
Admin.Segment
EDI.Collection
EDSD.Segment
Orders.config
UNCL.Codelist
3. Öffnen Sie die Datei **Orders-Custom-EDI.mfd** und klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



Wenn Sie auf die Schaltfläche "Ausgabe" klicken, wird das folgende Mapping-Ergebnis angezeigt.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3  <Header>
4      <Number>ABC123456XYZ</Number>
5  </Header>
6  <Customer>
7      <Number>92</Number>
8      <ContactName>Ted Little</ContactName>
9      <Salutation>Mr</Salutation>
10     <CompanyName>24h Pizza</CompanyName>
11     <Address>
12         <Street>Long Way</Street>
13         <City>San-Francisco</City>
14         <ZIP>34424</ZIP>
15         <State>CA</State>
16     </Address>
17 </Customer>
18 <Customer>
19     <Number>92</Number>
20     <ContactName>Michelle Butler</ContactName>
21     <Salutation>Mrs</Salutation>
22     <CompanyName>Nanonull, Inc.</CompanyName>
```

Anmerkung zur Codegenerierung:

Bei Generierung von C++-Code wird eine Klasse namens "CX1000Type" generiert, die über die Klasse "CC056Type" aufgerufen werden kann.

20.10 Anpassen einer ANSI X12-Transaktion

MapForce gestattet Ihnen, X12-Transaktionen an unterschiedliche, nicht-standardkonforme oder geänderte X12-Formate anzupassen.

In diesem Beispiel wird die Datei **Orders-Custom-X12.mfd** aus dem Ordner **...\MapForceExamples\Tutorial** verwendet. Beachten Sie bitte, dass der Ordner **..\Tutorial** auch die ZIP-Datei **X12.Nanonull.zip** enthält. Die ZIP-Datei enthält die Datei **result** in der das Ergebnis der Anpassung gespeichert ist.

Die oben erwähnte X12-Quelldatei **Orders-Custom.X12** im Ordner **..\Tutorial** wurde geändert und enthält nun ein neues Feld im Segment N2:

- Zeile 6 enthält den zusätzlichen Eintrag **++Mrs**

1	ISA+00+	+00+	+ZZ+SenderID	+ZZ+ReceiverID
2	GS+P0+SenderID+ReceiverID+20060308+182347+1+UN+050112'			
3	ST+850+12345'			
4	BEG+00+SA+ABC123456XYZ++20040430'			
5	N1+1 +Nanonull, Inc.++123'			
6	N2+Michelle Butler++Mrs'			
7	N3+119 Oakstreet Suite 4876'			
8	N4+Vereno+CA+29213'			
9	P01++1++7.2'			
10	P03+1 +++7.2++1+US'			
11	PID+1++++Pizza Pepperoni'			
12	TXI+1 ++9'			
13	AMT+1+720'			
14	P01++2++13.2'			
15	P03+1 +++6.6++2+US'			

Bitte beachten Sie:

Die Informationen zur Anpassung von EDIFACT und ANSI X12 befinden sich in zwei von der Struktur her identischen Dateien, die die Verwendung mehrerer aufeinander folgender Elemente desselben Namens in jedem der EDI-Formate erlauben. Für EDIFACT wird die Datei **EDSD.segment** verwendet, während für ANSI X12 die Datei **X12.Segment** verwendet wird. Nähere Informationen dazu finden Sie unter "[Mehrere aufeinander folgende Elemente](#)".

20.10.1 X12-Quelldateien und mehrere identische Felder

Beim Anpassen einer X12-Transaktion sollten Sie unbedingt beachten, dass in Segmenten oft mehrere aufeinander folgende Elemente desselben Namens verwendet werden können.

Mehrere aufeinander folgende Elemente desselben Namens

Ziel in diesem Abschnitt ist es, eine ANSI X12-Datei zu mappen, die an eine XML-Schemadatei angepasst wurde. Dem Segment N2 wurde das neue Feld "Mrs" hinzugefügt. Dieses zusätzliche Datenfeld soll auf das Feld "Salutation" in XML-Schema gemappt werden.

LOOP ID - N1		
3100	<u>N1</u>	<u>Party Identification</u>
3200	<u>N2</u>	<u>Additional Name Information</u>
3250	<u>IN2</u>	<u>Individual Name Structure Components</u>
3300	<u>N3</u>	<u>Party Location</u>
3400	<u>N4</u>	<u>Geographic Location</u>
3450	<u>NX2</u>	<u>Location ID Component</u>
3500	<u>REF</u>	<u>Reference Information</u>
3600	<u>PER</u>	<u>Administrative Communications Contact</u>

Um ein neues Feld zu einer X12-Datei hinzuzufügen, muss normalerweise ein einzelnes Trennzeichen, im Allgemeinen das +-Zeichen, gefolgt von den Daten hinzugefügt werden. Laut der unten gezeigten Segmentspezifikation für N2 sind zwei aufeinander folgende Felder mit der Bezeichnung "Name" zulässig.

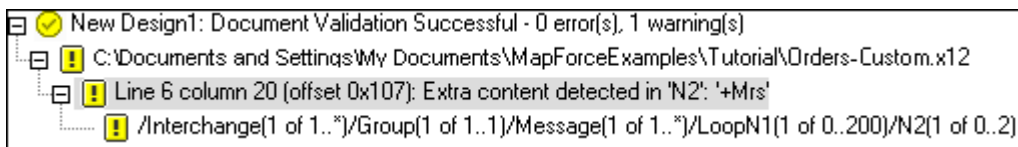
REF	ELE ID	NAME	RPT ATTRIBUTES
01	<u>93</u>	<u>Name</u>	M AN 1/60
02	<u>93</u>	<u>Name</u>	O AN 1/60

Dies muss daher in der X12-Quelldatei berücksichtigt werden. Fügen Sie dazu ein "leeres" Feld-Trennzeichen für den ersten (obligatorischen) Eintrag hinzu und einen zweiten, um die eigentlichen Daten zu trennen. Dies bedeutet, dass Sie **++Mrs** hinzufügen müssen, damit die Daten der X12-Spezifikation entsprechen.

In der mit MapForce, im Ordner **...MapForceEDI\X12** mitgelieferten Datei **X12.Segment** finden Sie Informationen darüber, welche Felder mehrere Einträge haben können. Alle Segmente, die mehrere identische Feldeinträge enthalten dürfen (z.B. 93), werden als Feldnummer mit einem Attribut **mergedEntries** angezeigt. Dieses Attribut definiert, wie viele Mehrfacheinträge zulässig sind, in diesem Fall 2.

```
<Segment name="N2" info="Additional Name Information">
  <Data ref="F93" mergedEntries="2"/>
</Segment>
```

Wenn in die X12-Datei nur ein +-Trennzeichen eingefügt wird, wird die folgende Warnmeldung im Meldungsfenster angezeigt, wenn die EDI-Komponente eingefügt wird und die EDI-Datei zugewiesen wurde.



Das Meldungsfenster enthält genaue Informationen zum Pfad der fehlerverursachenden Datei und der Ursache für den Fehler. Wenn Sie auf die jeweilige Meldung klicken, wird die jeweilige Zeile in der Komponente angezeigt, wenn ein Konnektor fehlt oder es wird angezeigt, dass für

eines der Datenelemente zusätzliche Daten vorhanden sind.

20.10.2 X12 Anpassung - Vorbereitung

In den folgenden Abschnitten wird beschrieben, wie die Konfigurationsdateien so angepasst werden können, dass die geänderte X12-Transaktion auf das Schema **ORDER-x12.xsd** gemappt werden kann. Es wird davon ausgegangen, dass die mitgelieferte ZIP-Datei **nicht** verwendet wird.

So bereiten Sie das Beispiel für die Anpassung vor:

1. Kopieren Sie die folgenden Dateien aus dem Ordner ...**MapForceEDI\X12** in den Ordner X12.Nanonull:
 EDI.Collection
 Envelope.Config
 850.Config
 X12.Segment
 X12.Codelist
2. Ändern Sie die Attribute dieser Dateien in **Lese/Schreibzugriff**, um sie editierbar zu machen.

Konfigurieren der Datei EDI.Collection:

1. Öffnen Sie die Datei "EDI.Collection" in XMLSpy oder im Editor Ihrer Wahl.
2. Entfernen Sie alle Nachrichtenelemente mit Ausnahme der "850" "Purchase Orders". Stellen Sie sicher, dass die <Messages> Tags erhalten bleiben!

```

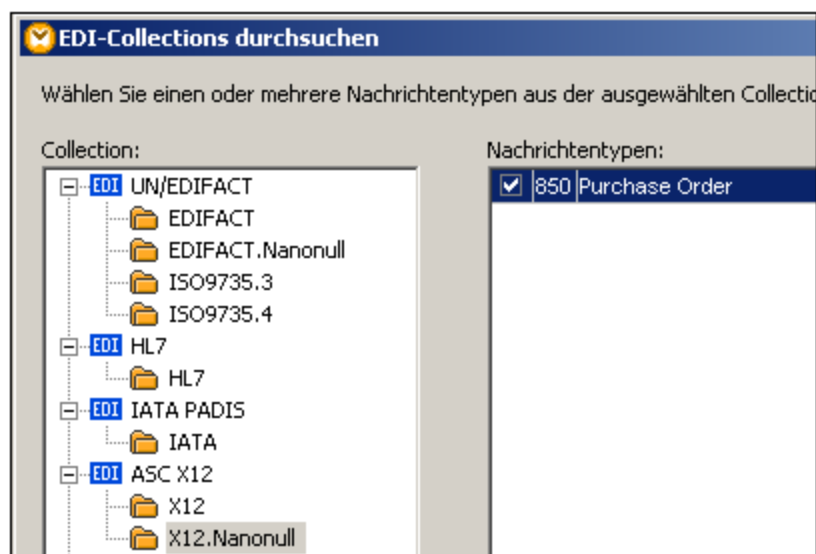
1      <?xml version="1.0" encoding="UTF-8"?>
2      <Messages Version="3">
3          <Meta>
4              <Release>6020</Release>
5              <Agency>X12</Agency>
6          </Meta>
7          <Root File="Envelope.Config"/>
8          <Message Type="850" File="850.Config" Description="Purchase Order"/>
9      </Messages>

```

3. Speichern Sie die Datei.

So sehen Sie den Inhalt der Collection-Datei:

- Starten Sie MapForce, wählen Sie **Einfügen | EDI** oder klicken Sie auf das Symbol "EDI-Datei einfügen". Daraufhin wird das Dialogfeld "EDI Collection durchsuchen" geöffnet. Es wird ein neuer Ordner "X12.Nanonull" angezeigt. Nur ein Eintrag ist zu sehen; die erste Spalte enthält den Message-Typ "850", die zweite den Text der Message-Beschreibung "Purchase Order".



Bitte beachten Sie:

MapForce durchsucht alle gleichrangigen Unterverzeichnisse im Verzeichnis "...\\MapforceEDI" nach einer Datei mit dem Namen "EDI.Collection". Der Ordnername, der eine EDI Collection enthält, wird als Ordner im Dialogfeld angezeigt. Im Listenfeld sehen Sie den aktuellen Inhalt der Collection-Datei, die in unserem Beispiel nur einen Purchase Order-Eintrag enthält.

Ziel dieses Abschnitts ist es, das N2-Segment "Additional Name Information" neu zu definieren, indem wir das Feld X1000 hinzufügen, um es für einzelne Transaktionen verfügbar zu machen. N2 besteht derzeit aus einem Feld, nämlich "F93 Name".

Es gibt verschiedene Methoden, wie Sie diese Anpassung vornehmen können:

- **Global** durch Anpassen der Datei **X12.Segment**. Alle Segmente in allen Transaktionen, die N2 verwenden, werden das neue Element enthalten/referenzieren.
- **Inline** durch Anpassen der Datei **850.Config**. Nur das angepasste Segment (N2) in der aktuellen Transaktion wird das neue Element enthalten.

EDI.Collection

Die Collection-Datei enthält eine Liste aller Transaktionen im aktuellen Verzeichnis. Sie dient dazu, eine Liste der verfügbaren Transaktionen zur Verfügung zu stellen, wenn eine X12-Datei in MapForce eingefügt wird. Die folgende Beispiel enthält nur eine Transaktion, nämlich "Purchase Order".

850.Config

Die Konfigurationsdatei für die Bestell-Transaktionsdateien. Diese Datei enthält alle Gruppen- und Segmentdefinitionen, die in Bestell-Transaktionen verwendet werden. Änderungen an dieser Datei können lokale oder inline-Anpassungen definieren.

X12.Segment

In dieser Datei werden die Namen der Segmente, Composites und Felder der X12-Dateien definiert. Sie wird beim Parsen der Datei verwendet. Änderungen an dieser Datei sind globale Anpassungen und gelten für alle Segmente und Transaktionen.

20.10.3 Globale Anpassung

- Änderungen müssen nur an der Datei **X12.Segment** vorgenommen werden, um globalen Zugriff auf das neue Feld X1000 zu haben.
 - Alle **Segmente** in allen **Transaktionen**, in denen Composite **N2** verwendet wird, werden daraufhin das neue Element bzw. eine Referenz darauf enthalten.

Redefinition in der Datei X12.Segment:

Öffnen Sie die Datei X12.Segment und navigieren Sie zu **Config | Elements | Segment name="N2" info="Additional Name Information"**.

```
<Segment name="N2" info="Additional Name Information">
  <Data ref="F93" mergedEntries="2"/>
</Segment>
```

Fügen Sie unterhalb von F93 die folgende Zeile ein und speichern Sie die Datei:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
```

Die Definition sieht aus wie unten gezeigt:

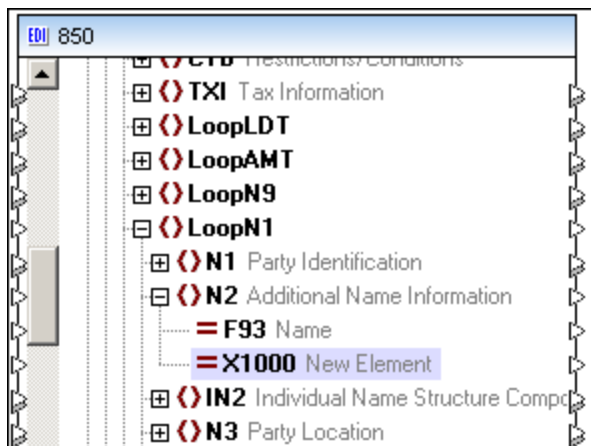
```
<Segment name="N2" info="Additional Name Information">
  <Data ref="F93" mergedEntries="2"/>
  <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
</Segment>
```

Bitte beachten Sie:

Das neue Feld X1000 wird im Gegensatz zu anderen Feldern, für die das "ref" Attribut verwendet wird, mit Hilfe des "name" Attributs definiert. Das Feld F93 wird am Beginn der X12-Datei mit Hilfe des Data name-Elements definiert und hier nur referenziert. Das neue Feld kann nun von verschiedenen Segmenten oder Composites aus referenziert werden.

Vorschau des neuen Felds in MapForce:

1. Wählen Sie die Menüoption **Einfügen | EDI**, oder klicken Sie auf das Symbol "EDI-Datei einfügen" .
2. Klicken Sie auf das Register "X12.Nanonull" und wählen Sie die 850 Purchase Order-Transaktion aus.
3. Klicken Sie auf die Schaltfläche "Überspringen", um die Auswahl der EDI-Quelldatei im Moment auszulassen.
Die 850-Komponente ist nun im Mapping-Fenster sichtbar.
4. Navigieren Sie zu **Envelope/Interchange/Group/Message_850/LoopN1/N2**, um das neue X1000 Element zu sehen.



20.10.4 Inline-Anpassung

- Änderungen müssen nur in der Datei **850.config** an der angegebenen Stelle vorgenommen werden, um lokalen Zugriff auf das neue Feld X1000 zu haben.
- Nur das neu definierte N2-Segment in der **aktuellen** Transaktion enthält/referenziert das neue X1000-Feld.
- Anders ausgedrückt, das Segment wird lokal neu definiert und enthält das neue Feld X1000.

Lokale Anpassung - Segment-Redefinition in der Datei 850.Config:

Öffnen Sie die Datei 850.Config und navigieren Sie zu **Config | Group | Message | Group name="LoopN1"** (oder suchen Sie nach **LoopN1**).

```

73 <Group name="LoopN1" maxOccurs="200" minOccurs="0">
74   <Segment ref="N1"/>
75   <Segment ref="N2" minOccurs="0" maxOccurs="2"/>
76   <Segment ref="IN2" minOccurs="0" maxOccurs="unbounded"/>
77   <Segment ref="N3" minOccurs="0" maxOccurs="2"/>
78   <Segment ref="N4" minOccurs="0" maxOccurs="unbounded"/>

```

Ersetzen Sie die Zeile Segment ref="N2"... durch die folgenden Zeilen:

```

<Segment name="N2" info="Additional Name Information">
  <Data ref="F93" mergedEntries="2"/>
  <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
</Segment>

```

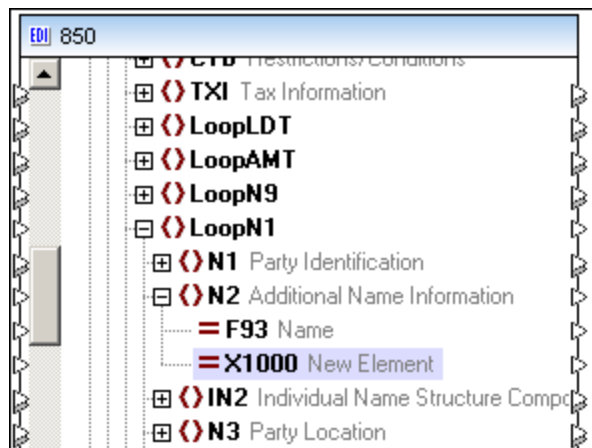
```

73 <Group name="LoopN1" maxOccurs="200" minOccurs="0">
74   <Segment ref="N1"/>
75   <Segment name="N2" info="Additional Name Information">
76     <Data ref="F93" mergedEntries="2"/>
77     <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
78   </Segment>
79   <Segment ref="IN2" minOccurs="0" maxOccurs="unbounded"/>

```

Vorschau des neuen Felds in MapForce:

1. Wählen Sie die Menüoption **Einfügen | EDI**, oder klicken Sie auf das Symbol . "EDI-Datei einfügen".
2. Klicken Sie auf das Register "X12.Nanonull" und wählen Sie die 850 Purchase Order-Transaktion aus.
3. Klicken Sie auf die Schaltfläche "Abbrechen", um die Auswahl der EDI-Quelldatei im Moment auszulassen. Die 850-Komponente ist nun im Mapping-Fenster sichtbar.
4. Navigieren Sie zu **Envelope/Interchange/Group/Message/LoopN1/N2**, um das neue X1000 Element zu sehen.

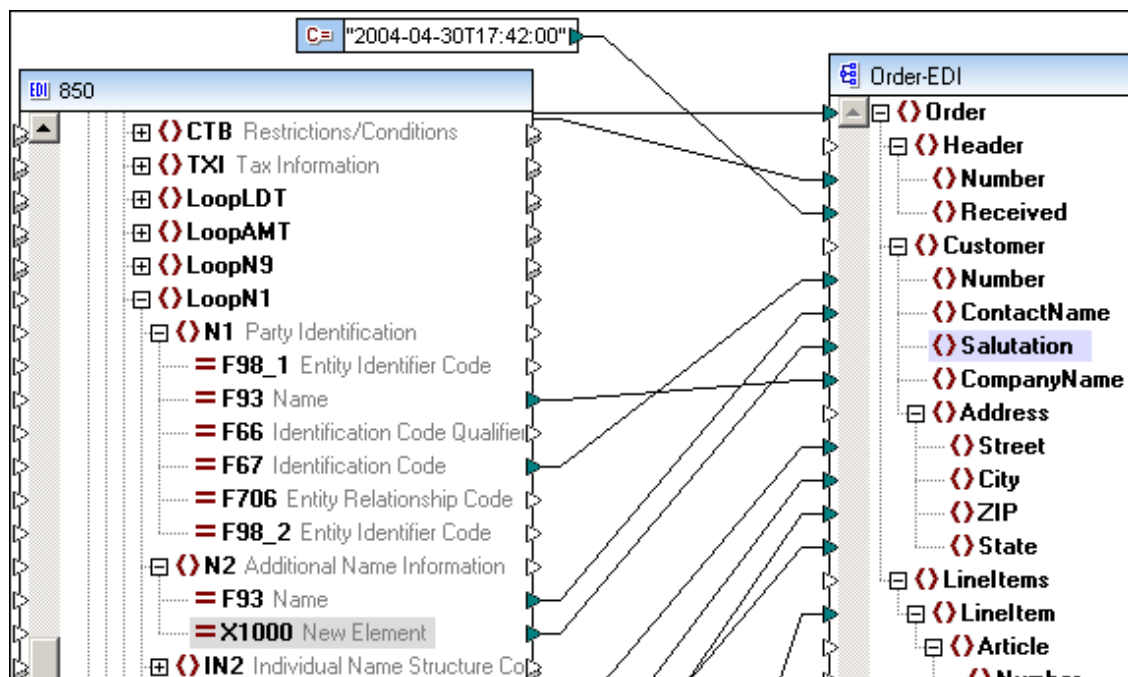


20.10.5 X12-Mapping-Beispiel

Das in den Abbildungen unten gezeigte Mapping **Orders-Custom-X12.mfd** steht im Verzeichnis [...\MapForceExamples\Tutorial\](#) zur Verfügung.

In dem Beispiel wird die Datei **Orders-Custom.x12** auf das Schema **Order-X12** gemappt. Das Feld **X1000**, das zu EDI-Struktur hinzugefügt wurde, wurde auf das Datenelement "Salutation" gemappt.

1. Erstellen Sie im Verzeichnis **"...\\MapforceEDI"** einen neuen Ordner und geben Sie ihm z.B. den Namen **"X12.Nanonull"**
2. Entpacken Sie die mitgelieferte Datei **X12.Nanonull.zip** (aus dem Ordner **...\Tutorial**) in den neuen Ordner.
3. Öffnen Sie die Datei **Orders-Custom-X12.mfd**.



Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie das unten gezeigte Mapping-Ergebnis.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3  <Header>
4      <Number>ABC123456XYZ</Number>
5      <Received>2004-04-30T17:42:00</Received>
6  </Header>
7  <Customer>
8      <Number>123</Number>
9      <ContactName>Michelle Butler</ContactName>
10     <Salutation>Mrs</Salutation>
11     <CompanyName>Nanonull, Inc.</CompanyName>
12     <Address>
13         <Street>119 Oakstreet Suite 4876</Street>
14         <City>Vereno</City>
15         <ZIP>29213</ZIP>
16         <State>CA</State>
17     </Address>

```

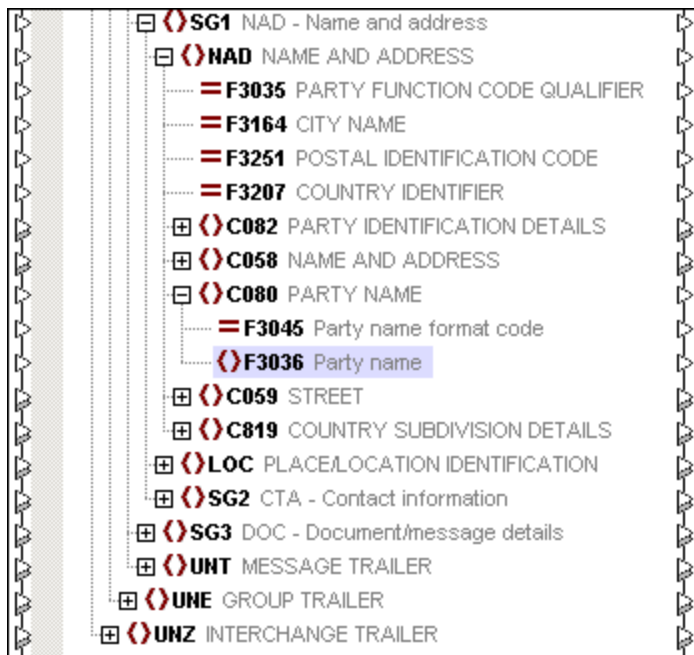
20.11 Aufgliedern zusammengeführter Einträge in separate Nodes/Datenelemente

Beim Mappen von EDI-Komponenten liefert MapForce unter Umständen automatisch einen Feldeintrag als ein **einziges** mapbares Datenelement / einen Node in der EDI-Komponente, auch wenn es/er aus mehreren aufeinander folgenden Instanzen desselben Felds besteht.

Informationen über die Anpassung von EDIFACT und ANSI X12 werden in zwei von ihrer Struktur her identischen Dateien bereit gestellt, mit Hilfe derer mehrere aufeinander folgende Elemente desselben Namens in den einzelnen EDI-Formaten verwendet werden können. In EDIFACT wird die Datei **EDSD.Segment** verwendet, während in ANSI X12 die Datei **X12.Segment** verwendet wird.

Durch Bearbeitung der Datei "EDSD.Segment" können Sie die zusammengeführte Sequenz in ihre Einzelteile (Instanzen) aufgliedern und eindeutige Nodes/mapbare Datenelemente erstellen. Erstellen Sie bitte unbedingt ein neues Verzeichnis für die Datei "EDSD.Segment" und andere EDI-Dateien, die Sie eventuell bearbeiten, wie [hier](#) beschrieben.

Die unten gezeigte OSTRPT-Komponente enthält das Feld "F3036 Party name", welches als einziges mapbares Datenelement gezeigt wird. Das Feld "Party name" kann jedoch fünf separate/eindeutige Party Names pro EDIFACT-Definition enthalten.



Wenn Sie die Datei "EDSD.Segment" öffnen, sehen Sie die Definition des zusammengesetzten C080 Party Name. Ein Segment, das mehrere identische Feldeinträge enthält (z.B. F3036), wird als Feldnummer mit einem **mergedEntries**-Attribut angezeigt, welches definiert, wie viele Instanzen vorkommen dürfen (z.B. 5 wie in der Abbildung unten).

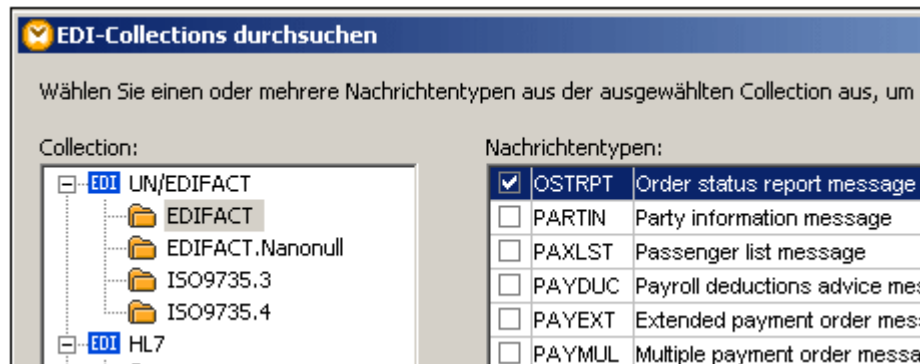
```
<Composite name="C080" info="PARTY NAME">
  <Data ref="F3036" mergedEntries="5"/>
  <Data ref="F3045" minOccurs="0"/>
</Composite>
```

So erstellen Sie anhand einer zusammengeführten Sequenz mehrere eindeutige Felder:

1. Öffnen Sie die Datei **EDSD.Segment** (X12.Segment) aus dem Ordner c:\Program Files\Altova\MapForce2012\MapForceEDI\EDIFACT\ (oder c:\Program Files\Altova\MapForce2012\MapForceEDI\X12\)
2. Entfernen Sie das Attribut **mergedEntries="5"**.
3. Kopieren Sie das restliche <Data ref="F3036"/> Element und fügen Sie es so oft ein, wie im mergedEntries-Attribut angegeben, also 5 Mal.
4. Fügen Sie zu allen Feldern mit Ausnahme des ersten das Attribut **minOccurs="0"** hinzu.
5. Geben Sie für jedes der Felder einen eindeutigen Node-Namen ein, z.B. **nodeName_1** usw.

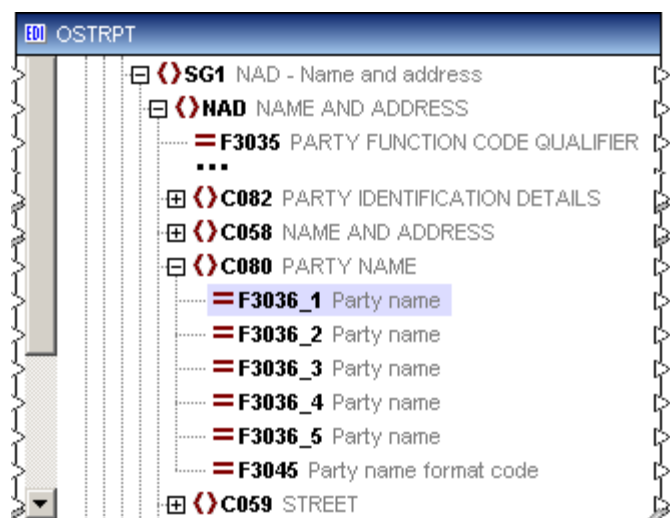
```
<Composite name="C080" info="PARTY NAME">
  <Data ref="F3036" nodeName="F3036_1"/>
  <Data ref="F3036" nodeName="F3036_2" minOccurs="0"/>
  <Data ref="F3036" nodeName="F3036_3" minOccurs="0"/>
  <Data ref="F3036" nodeName="F3036_4" minOccurs="0"/>
  <Data ref="F3036" nodeName="F3036_5" minOccurs="0"/>
  <Data ref="F3045" minOccurs="0"/>
</Composite>
```

6. Speichern Sie die bearbeitete EDSD.Segment-Datei.
7. Öffnen Sie MapForce, klicken Sie auf die Schaltfläche "EDI einfügen"  und fügen Sie die "Order Status Report Message" aus dem Register EDIFACT ein.



Daraufhin wird die OSTRPT-Komponente erstellt.

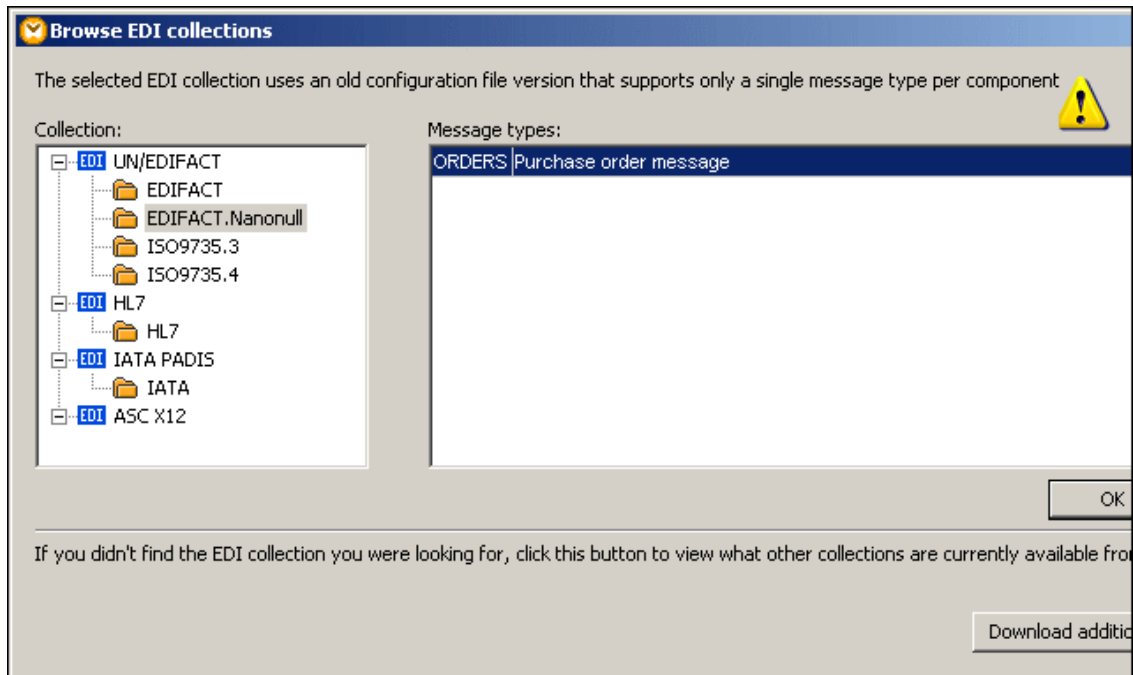
8. Erweitern Sie die Datenelemente bis **Envelope | Interchange | Group | Message_OSTRPT | SG1 NAD**.
9. Erweitern Sie C080, um die neuen eindeutigen Party name-Felder zu sehen. Sie können nun Einträge von und auf diese einzelnen Party name-Datenelemente/Nodes mappen.



Nähere Informationen dazu finden Sie unter "[Mehrere aufeinander folgende Elemente](#)".

20.12 Aktualisieren von config-Dateien zur Unterstützung mehrerer Nachrichten

Wenn Sie EDI-Konfigurationsdateien, die älter als Version 3 sind, verwenden, so werden Sie im unten gezeigten Dialogfeld aufmerksam gemacht, dass Sie eine Konfigurationsdatei verwenden, die nur einen Nachrichtentyp pro Komponente unterstützt.



So aktualisieren Sie die Konfigurationsdateien, damit mehrere Nachrichtentypen pro Komponente unterstützt werden:

1. Kopieren Sie Envelope.Config aus dem Original config-Ordner (z.B. EDIFACT) in den Ordner, der die angepassten config-Dateien enthält (z.B. EDIFACT.Nanonull).
2. Bearbeiten Sie **EDI.Collection**, ändern Sie das Version-Attribut des Root-Elements von 2 in 3.
3. Fügen Sie den Tag "<Root File='Envelope.Config'/" hinter dem </Meta>-Tag hinzu:

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="3">
  <Meta>
    <Version>D</Version>
    <Release>04B</Release>
    <Agency>UN</Agency>
  </Meta>
  <Root File="Envelope.Config"/>
  <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase
order message"/>
</Messages>
```

4. Bearbeiten Sie **ORDERS.Config**, ändern Sie das Version-Attribut des Root-Elements von 2 in 3.
5. Fügen Sie "<Format standard='EDIFACT'/" (oder X12 oder HL7) hinzu.

- ```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="3">
 <Meta>
 <Version>D</Version>
 <Release>04B</Release>
 <Agency>UN</Agency>
 </Meta>
 <Format standard="EDIFACT"/>
 <Include href="Admin.Segment"/>
 <Include href="EDSD.Segment"/>
 <Include href="UNCL.Codelist"/>
 <Message>
 <MessageType>ORDERS</MessageType>
 <Description>Purchase order message</Description>
 <Revision>14</Revision>
 <Date>2004-11-23</Date>
 <Group name="Envelope">
 <Group name="Interchange" maxOccurs="unbounded">
 <Segment ref="UNA" minOccurs="0"/>
 <Segment ref="UNB" minOccurs="0"/>
 <Group name="Group"
maxOccurs="unbounded">
 <Segment ref="UNG"
minOccurs="0"/>
 <Group name="Message_ORDERS"
maxOccurs="unbounded" info="UNH - Message header">
 <Segment ref="UNH"/>
 <Segment ref="BGM"/>
 .
 <Segment ref="UNT"/>
 </Group>
 <Segment ref="UNE" minOccurs="0"/>
 </Group>
 </Group>
 </Message>
```

Gehen Sie ebenso vor, um die EDI X12- oder HL7-Konfigurationsdateien zu aktualisieren.

## 20.13 SAP IDOCS

SAP IDocs (intermediate documents) Dokumente dienen zum Austauschen von Geschäftsdaten zwischen SAP und Nicht-SAP-Applikationen. Bei den Dokumenten handelt es sich um eine Form von Datenzwischenspeicher, das zwischen unterschiedlichen Systemen ausgetauscht werden kann.

Ein IDoc hat die folgende Struktur:

**Kontrollsatz:** Enthält Kontrollinformationen über das IDoc: Absender, Empfänger, Nachrichtentyp und IDoc-Typ. Das Kontrollsatzformat ist für alle IDoc-Typen ähnlich.

**Datensegment:** Enthält die eigentlichen Daten des Segments sowie andere Metadaten: Überschrift, Segmentnummer und Typ sowie die Felder, die die Daten enthalten.

**Statussätze:** Enthalten Informationen zum aktuellen Status des Dokuments, d.h. die derzeit verarbeiteten Schritte und die Schritte, die noch verarbeitet werden müssen. Das Statusformat ist für alle IDoc-Typen identisch.

Die Versionsnummer in der Port-Definition definiert die Systeme, mit denen kommuniziert wird. Die Hauptunterschiede zwischen den Versionen sind die verschiedenen Namenslängen, die in verschiedenen Elementen verwendet werden, und die Verwendung von Erweiterungen. SAP R3 Version 4.X unterstützt lange Namen (sowie Erweiterungen), während die früheren Versionen dies nicht tun.

Port Version 1: Releases 2.1. und 2.2.

Port Version 2: Releases 3.0, 3.1 und R/2 Systeme.

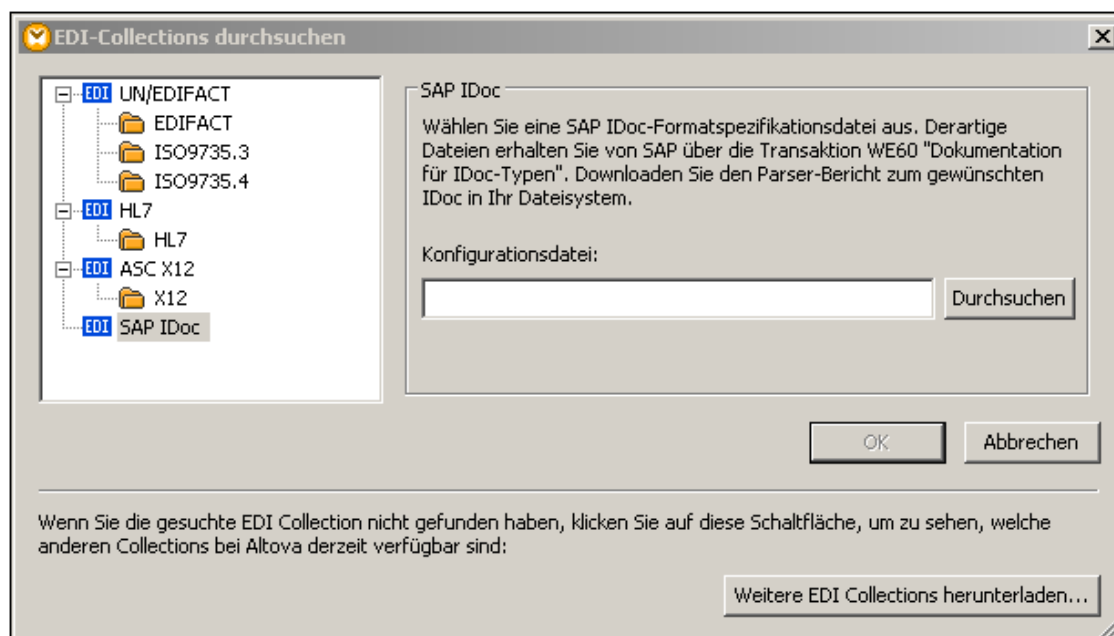
Port Version 3: Release 4.x (Standardwert)

MapForce behandelt IDoc-Komponenten als Dateien mit einer festen Länge von 30 Zeichen für den Message Typ, 30 für den IDoc Typ und 27 für Segmentfelder.

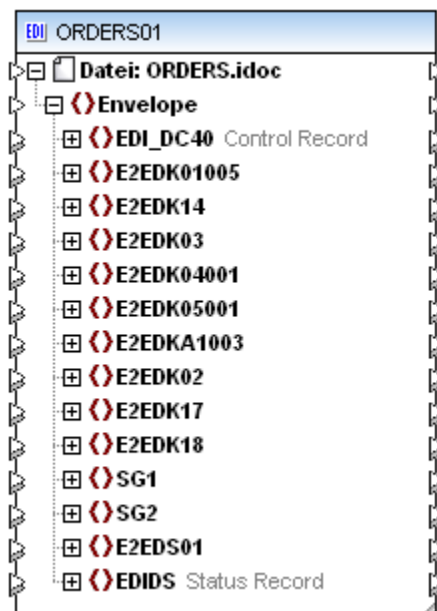
In der Datei **IDoc\_Order.mfd** im Ordner ...MapForceExamples sehen Sie ein Beispiel für das Mapping einer SAP IDoc-Datei auf eine XML-Schema-Zieldatei.

### So fügen Sie ein SAP IDoc-Dokument ein:

1. Wählen Sie die Menüoption **Einfügen | EDI**.  
Daraufhin wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet.
2. Klicken Sie im Listenfeld auf den Eintrag **SAP IDoc**.
3. Klicken Sie auf die Schaltfläche **Durchsuchen**, um die Konfigurationsdatei auszuwählen und anschließend zum Fortfahren auf **OK**.  
Sie erhalten diese Konfigurationsdatei über die Transaktion WE60 "Documentation for IDoc types" von Ihrem SAP-System. Im Ordner ...MapForceExamples finden Sie eine Konfigurationsdatei ORDERS01-Parseridoc.txt für das Beispiel **IDoc\_Order.mfd**.



4. Klicken Sie im Meldungsfeld auf die Schaltfläche "Durchsuchen", um eine EDI-Beispieldatei (\*.idoc) auszuwählen, aus der bei Erstellung einer Quellkomponente die Daten genommen werden. ORDERS.idoc finden Sie im Ordner ...MapForceExamples. Daraufhin wird das Dialogfeld "Komponenteneinstellungen" angezeigt.
5. Klicken Sie auf OK, um das Dialogfeld "Komponenteneinstellungen" zu schließen und die IDoc-Komponente einzufügen.



Nähere Informationen zum Aufrufen der Transaction WE60 und zum Generieren des Parser-Berichts finden Sie in der SAP IDoc-Dokumentation.

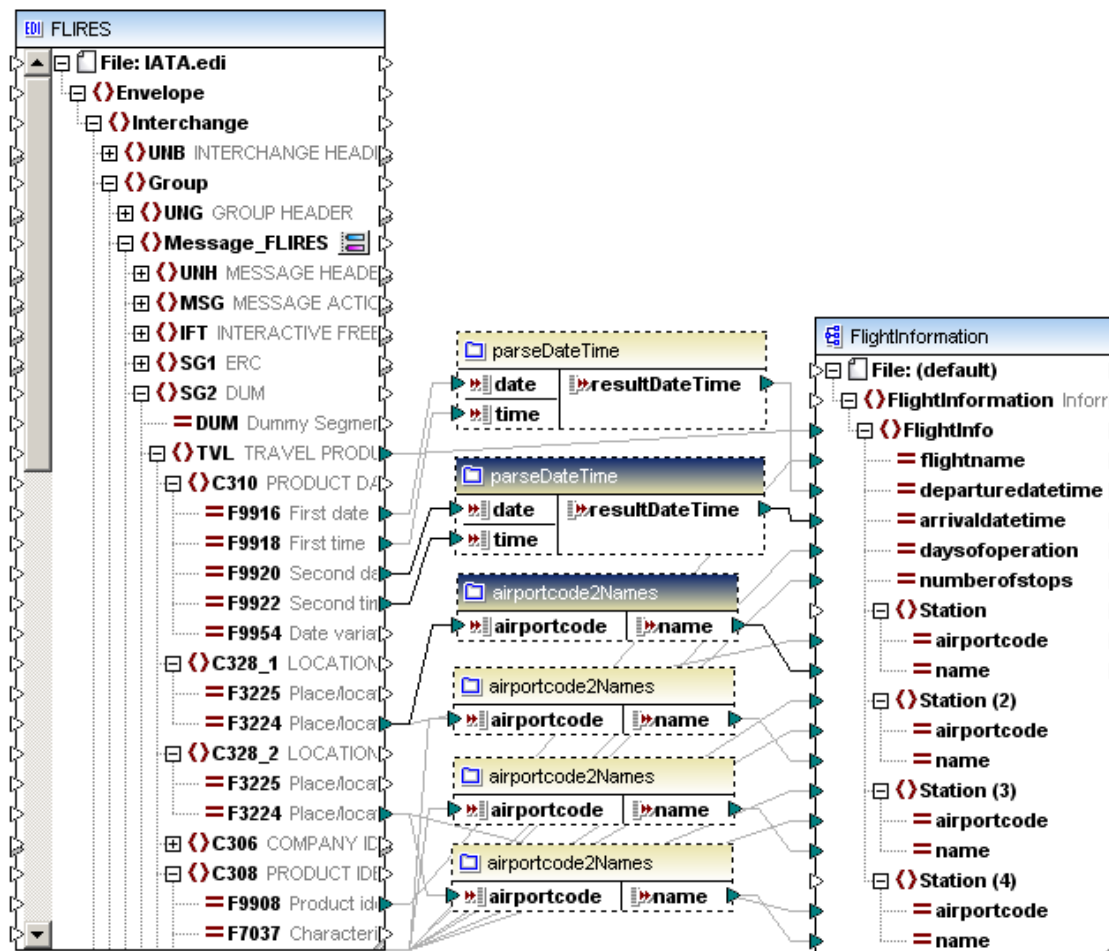


## 20.14 IATA PADIS

PADIS (Passenger and Airport Data Interchange Standards) ist eine Gruppe von Nachrichten, für die die EDIFACT (ISO 9735) Syntax verwendet wird.

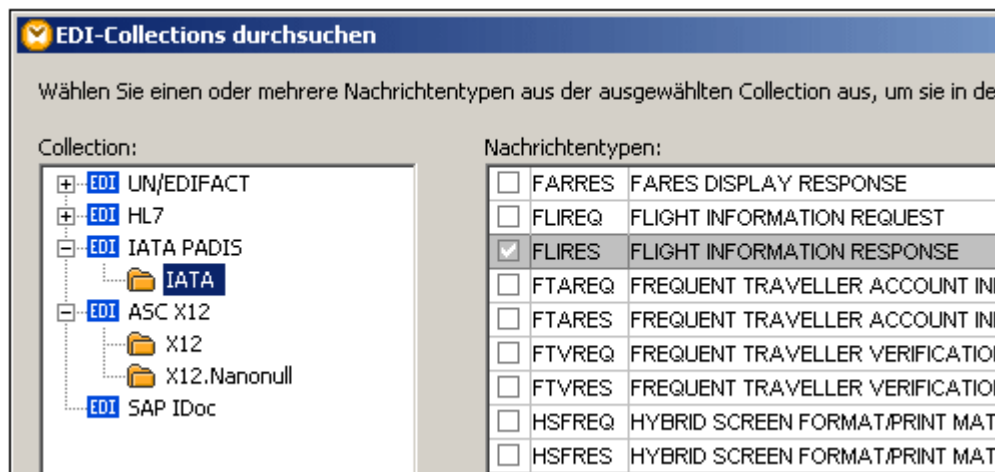
- MapForce unterstützt derzeit nur Nachrichten, in denen die UNH/UNT Message Header und Trailer-Segmente verwendet werden.
- MapForce unterstützt die Collections IATA.00.1 bis IATA.08.1.

In der Datei **IATA\_FlightInformationReport.mfd** im Ordner [...MapForceExamples](#) sehen Sie ein Beispiel für ein Mapping einer IATA PADIS-Datei auf eine XML-Schema-Datei.

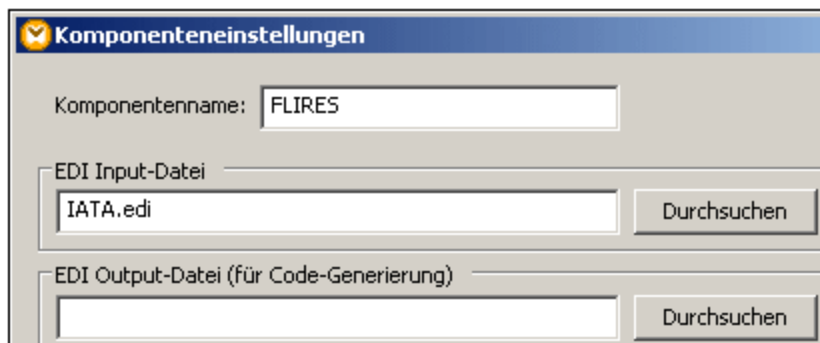


### So fügen Sie eine PADIS-Komponente ein:

1. Wählen Sie die Menüoption **Einfügen | EDI**.  
Daraufhin wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet.
2. Klicken Sie im Listenfeld "Collections" unter dem Eintrag IATA PADIS auf den Ordner IATA (oder auf die gewünschte IATA-Collection).
3. Aktivieren Sie das/die Kontrollkästchen des/der Nachrichtentypen, der/die eingefügt werden sollen, z.B. FLIRES und klicken Sie auf OK.



4. Klicken Sie in der daraufhin angezeigten Aufforderung auf die Schaltfläche "Durchsuchen" um eine EDI-Beispieldatei anzugeben, z.B. IATA.edi.  
Daraufhin wird das Dialogfeld "Komponenteneinstellungen" angezeigt.



Die IATA.edi-Datei liefert die Daten für die Quellkomponente. IATA.edi befindet sich im Ordner [...MapForceExamples](#).

5. Klicken Sie auf OK, um das Dialogfeld "Komponenteneinstellungen" zu schließen und fügen Sie die PADIS-Komponente ein.



Bitte beachten Sie:

Wenn Sie auf die Schaltfläche "Wählen Sie EDI-Nachrichtentypen aus"  klicken, wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet, in dem Sie mehrere Nachrichten pro Nachrichtenaustausch (pro EDI-Komponente) wählen können.

## 20.15 HIPAA X12

HIPAA X12 ist die neueste Version des Standards für elektronische Gesundheitsdaten, der vom US Department of Health and Human Services basierend auf EDI X12 Version 5010 zum Zweck der elektronischen Übertragung medizinischer Daten zwischen Versicherern, Gesundheitsdienstleistern und Arbeitgebern ins Leben gerufen wurde.

MapForce unterstützt die neueste Version, A2, der HIPAA-Implementierungsspezifikation (TR3). Ältere Versionen können in Form separater ZIP-Dateien von der [Altova Website](#) heruntergeladen werden.

HIPAA-Komponenten ähneln gewöhnlichen ANSI X12-Komponenten und MapForce unterstützt die folgenden Transactions:

|          |                                                                                 |
|----------|---------------------------------------------------------------------------------|
| X12 name | Nachrichtenname                                                                 |
| X279A1   | "Health Care Eligibility Benefit Inquiry (270)"                                 |
| X279A1   | "Health Care Eligibility Benefit Response (271)"                                |
| X212     | "Health Care Claim Status Request (276)"                                        |
| X212     | "Health Care Information Status Notification (277)"                             |
| X214     | "Health Care Claim Acknowledgment (277)"                                        |
| X217     | "Health Care Services Review - Request for Review (278)"                        |
| X217     | "Health Care Services Review - Response (278)"                                  |
| X218     | "Payroll Deducted and Other Group Premium Payment for Insurance Products (820)" |
| X220A1   | "Benefit Enrollment and Maintenance (834)"                                      |
| X221A1   | "Health Care Claim Payment/Advice (835)"                                        |
| X222A1   | "Health Care Claim: Professional (837)"                                         |
| X224A2   | "Health Care Claim: Dental (837)"                                               |
| X223A2   | "Health Care Claim: Institutional (837)"                                        |
| X231A1   | "Implementation Acknowledgment For Health Care Insurance (999)"                 |

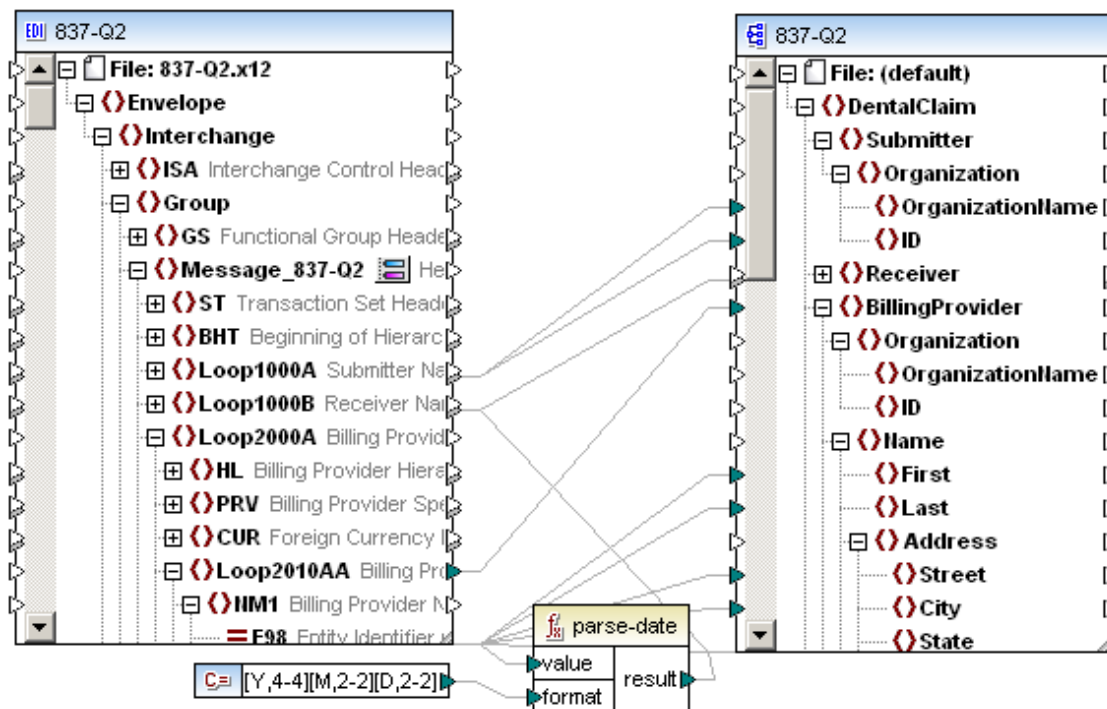
Unterschiede zur Behandlung von Standard X12-Nachrichten in MapForce:

- Die Hierarchie von HL-Segmenten wird beibehalten.
- So genannte "floating structures" (in 837-Nachrichten) werden unterstützt.
- MapForce komplettiert und validiert automatisch mehrere Felder.

MapForce unterstützt außerdem die automatische Generierung des [X12 999 Implementation Acknowledgment](#), das dem bestehenden 999- und 997-Functional Acknowledgment im X12-Standard ähnlich ist.

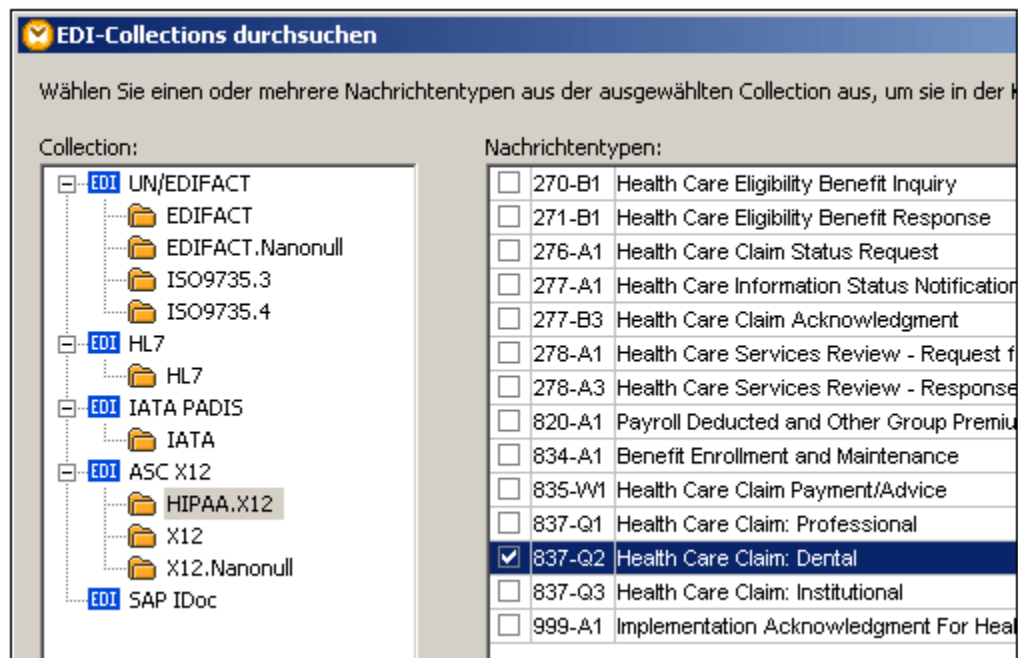
Außerdem unterstützt MapForce bei HIPAA-Komponenten mehrere Nachrichten pro Nachrichtenaustausch (d.h. EDI-Komponente).

Die Datei **HIPAA\_837D.mfd** im Ordner ...MapForceExamples enthält ein Beispiel für ein Mapping eines 837-Q2 Health Care Claim: Mapping einer Datei mit zahnärztlichen Daten auf eine XML-Schema-Zieldatei.



### So fügen Sie eine HIPAA X12-Komponente ein:

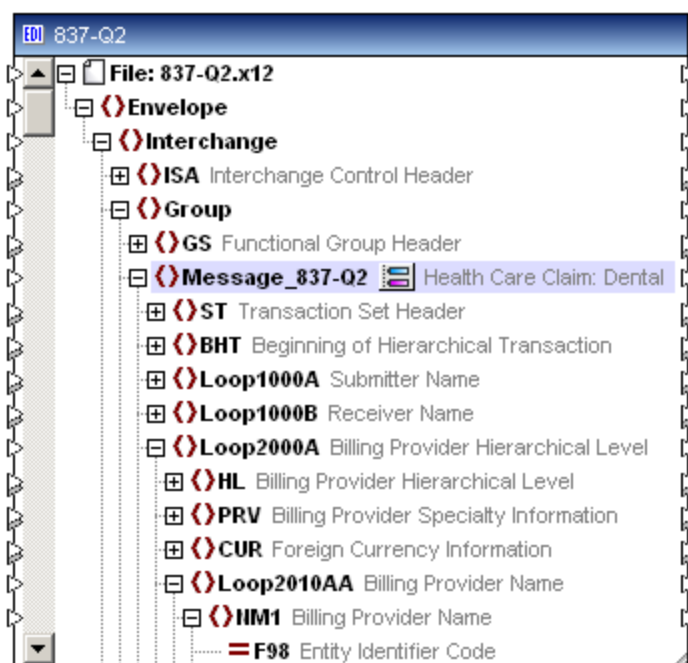
1. Wählen Sie die Menüoption **Einfügen | EDI**.  
Daraufhin wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet.
2. Klicken Sie im Listenfeld "Collection" unter dem Eintrag "ASC X12" auf den Ordner "HIPAA.X12" (oder die jeweilige HIPAA Collection, die Sie benötigen).
3. Aktivieren Sie das Kontrollkästchen für den Nachrichtentyp / die Nachrichtentypen, den / die Sie einfügen möchten, z.B. 837-Q2: Health Care Claim: Dental und klicken Sie anschließend auf OK.



4. Klicken Sie in der folgenden Eingabeaufforderung auf die Schaltfläche "Durchsuchen", um eine EDI-Beispieldatei anzugeben, z.B. 837-Q2.x12. Daraufhin wird das Dialogfeld "Komponenteneinstellungen" angezeigt. (Sie können die Trennzeichen zu diesem Zeitpunkt anpassen, falls nötig.)

Diese Datei liefert die Daten für die Quellkomponente. 837-Q2.x12 befindet sich im Ordner ...MapForceExamples.

5. Klicken Sie auf OK, um das Dialogfeld "Komponenteneinstellungen" zu schließen und die HIPAA-Komponente einzufügen.

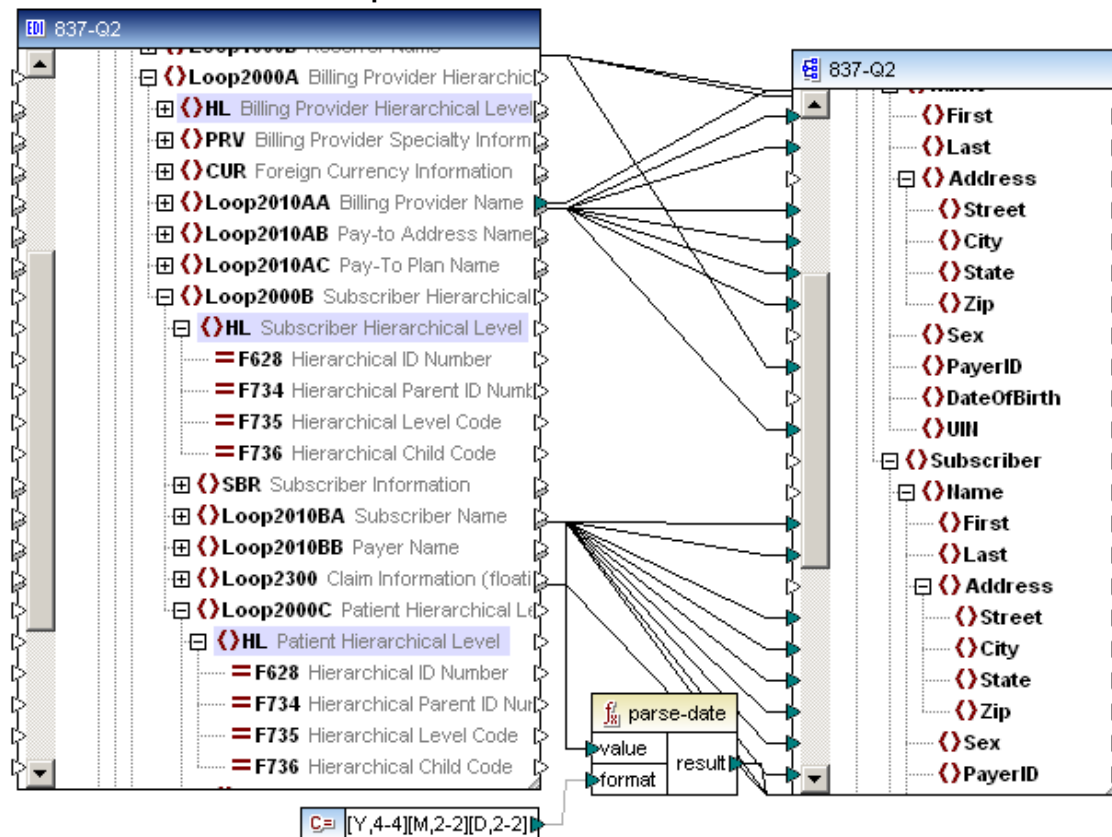


Bitte beachten Sie:

Mit der Schaltfläche "EDI-Nachricht auswählen"  wird das Dialogfeld "EDI-Collections durchsuchen" geöffnet, in dem Sie mehrere Nachrichten pro Austausch (pro EDI-Komponente) auswählen können.

## 20.15.1 HIPAA Transactions

### 837 Dental Transaction in MapForce



#### HL-Segmenthierarchie

HL-Segmente werden automatisch generiert und müssen vom Benutzer nicht manuell gemappt werden. Allerdings können bei Bedarf Werte gemappt werden.

#### Autokomplettierung

Die Konfigurationsdateien für HIPAA-Transactions enthalten zahlreiche Codes und werden, wo möglich, zur Autokomplettierung von Feldern verwendet. Die Autokomplettierung ist kontextsensitiv, daher können Felder an verschiedenen Stellen unterschiedliche Werte haben.

Die Autokomplettierungsfunktion kann durch Deaktivieren des Kontrollkästchens "Autokomplettierung von fehlenden Feldern" im Dialogfeld "Komponenteneinstellungen" ausgeschaltet werden.

#### Validierung

Die Validierung ist der Autokomplettierung insofern ähnlich, als sie ebenfalls kontextsensitiv erfolgt. Validierungsfehler werden auf dieselbe Art wie bei der [X12-Validierung](#) behandelt.

#### Bearbeiten von Konfigurationsdateien

HIPAA-Konfigurationsdateien ähneln X12-Konfigurationsdateien, weisen aber zusätzlich folgende Unterschiede auf:

- Werte (Codeliste)
- Bedingungen
- Komplettierungs-Flags

- "Not Used" Datenelemente

Feldinstanzen können eine integrierte Codeliste haben, die zur Validierung und/oder Autokomplettierung herangezogen wird. Werte können manuell hinzugefügt oder entfernt werden.

```
<Data ref="F365" info="Communication Number Qualifier" nodeName="F365_1">
 <Values>
 <Value Code="EM" />
 <Value Code="FX" />
 <Value Code="TE" />
 </Values>
</Data>
```

### Bedingungen

Bedingungen werden in Verbindung mit Werten angewendet. Da ein Segment oder eine Gruppe (Schleife) mehrmals vorkommen kann, werden diese Instanzen durch eine spezielle Bedingung identifiziert. Damit eine Bedingung erfüllt wird, muss ein Wert vorhanden sein, sonst wird die spezifische Gruppe nicht gefunden.

So kommt z.B. Loop1000 mehrmals vor, doch hat jede Instanz davon eine andere semantische Bedeutung.

Um zu erkennen, um welche Schleife es sich handelt, definiert MapForce mit Hilfe einer Bedingung, dass Loop1000A im Feld F98 von Segment NM1 einen Wertcode "41" haben muss. Wenn dies nicht der Fall ist, wird der Gruppenname Loop1000A nicht gefunden.

```
<Group name="Loop1000A" info="Submitter Name">
 <Segment name="NM1" info="Submitter Name">
 <Condition path="F98" />
 <Data ref="F98" info="Entity Identifier Code">
 <Values>
 <Value Code="41" />
 </Values>
 </Data>
 </Segment>
</Group>
(...)
```

Bedingungscode werden in der Zielkomponente automatisch generiert, wenn sie einen einzigen Wert enthalten.

Konfigurationsdateien werden mit einer Methode generiert, die gewährleistet, dass Bedingungswerte innerhalb der Sequenz sich wiederholender Segmente/Schleifen garantiert eindeutig sind.

Wenn Bedingungen oder ihre Werte bearbeitet werden müssen, muss diese Voraussetzung der Eindeutigkeit unbedingt berücksichtigt werden.

### Komplettierungs-Flags

Die Autokomplettierung kann auf Ebene der Konfigurationsdatei angepasst werden. Ein neues Element "Completion" mit drei Attributen wurde zur Definition der folgenden Dinge hinzugefügt:

- singleConditions – Einzelbedingungen für alle Felder in der Zielkomponente automatisch komplettieren
- singleValues – Einzelwerte in der Zielkomponente für alle Felder automatisch komplettieren
- HL – entsprechende Felder (für alle HL-Segmente) in der Zielkomponente im HL7-Segment generieren



(wobei "1" für "true" steht)

```
<Message>
<MessageType>837-Q2</MessageType>
<Completion singleConditions="1" singleValues="1" HL="1" />
<Description>Health Care Claim: Dental</Description>
(...)
```

#### **“Not Used” Datenelemente**

In HIPAA wird eine Reihe optionaler Elemente, die in Standard X12 Transactions vorhanden sind, weggelassen. Diese optionalen Elemente werden in MapForce-Komponenten versteckt.

Der Validierungsprozessor überprüft diese nicht verwendeten Felder, um sicher zu stellen, dass diese in Quelldateien nicht vorhanden sind. Da diese Felder nicht auf die Zielkomponente gemappt werden können (da sie versteckt werden), können sie auch nicht in Output-Zieldateien aufscheinen.

Für diese Felder wird in den Konfigurationsdateien das maxOccurs-Attribut "0" definiert. Falls nötig, können Sie mit Hilfe des maxOccurs-Attributs bestimmte Felder manuell verbergen (oder wieder einblenden).

```
<Data ref="F1037" minOccurs="0" info="Submitter Middle Name or Initial" />
<Data ref="F1038" minOccurs="0" maxOccurs="0" />
<Data ref="F1039" minOccurs="0" maxOccurs="0" />
```

#### **Geltungsbereich der Validierung**

Eine semantische Validierung wird von MapForce nicht unterstützt. Dies bedeutet, dass "Situationsfelder" einfach als optional behandelt werden.



# Kapitel 21

---

Mappen von HL7 v3.x von/auf XML-Schemas

## 21 Mappen von HL7 v3.x von/auf XML-Schemas

MapForce 2012 unterstützt automatisch HL7 Version **3.x.**, da diese auf XML basiert.

Auf der Seite [MapForce-Bibliotheken](#) der Altova-Website finden Sie einen separaten Installer für die HL7 V2.2 - V2.5.1 XML-Schemas und Konfigurationsdateien. Wählen Sie im Installer die benutzerdefinierte Installation aus, um nur die HL7 V3-Komponenten und XML-Schemas zu installieren.

Pfad der HL7 XML-Schemas nach der Installation:

|                       |                                                    |
|-----------------------|----------------------------------------------------|
| Windows XP-System:    | "C:\Program Files\Altova\Common2012\Schemas\hl7v3" |
| Windows Vista-System: | "C:\Program Files\Altova\Common2012\Schemas\hl7v3" |
| Windows7-System:      | "C:\Program Files\Altova\Common2012\Schemas\hl7v3" |

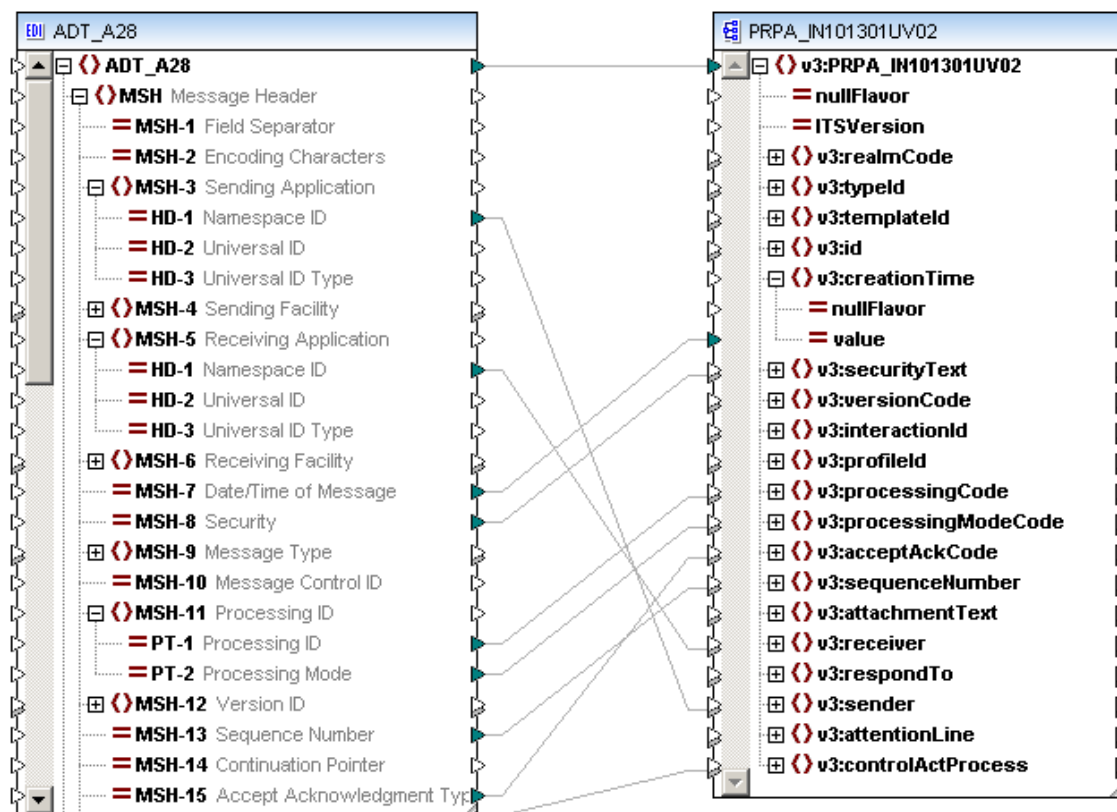
Wenn eine 32-Bit-MapForce Applikation auf einem 64-Bit-Betriebssystem verwendet wird, so lautet der Pfad

"C:\Program Files(x86)\Altova\Common2011\Schemas\hl7v3".

HL7-Dokumente können in MapForce als Quell- und Zielkomponenten verwendet werden. Diese Daten können auf beliebig viele XML-Schema-, Datenbank-, EDI- oder andere Komponenten gemappt werden.

In diesem Beispiel HL7V260\_To\_HL7V3.mfd aus dem Ordner ...MapForceExamples wird ein Teil eines HL7 V2.6-Dokuments auf ein XML-basiertes HL7 V3 Dokument gemappt.

- Die EDI-Datei ADT\_A28 finden Sie im Ordner ...MapForceExamples.
- Die XSD-Zieldatei PRPA\_IN101301UV02.xsd befindet sich im Ordner ...  
**MapForceExamples\HL7V3\_Example\_Schemas.**



Unten sehen Sie die erzeugte XML-Instanzdatei.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <PRPA_IN101301UV02 xsi:schemaLocation="urn:hl7-org:v3
 C:/DOCUME~1/MY/MYDOCU~1/Altova/MapForce2009/MapForceExamples/HL7V3_Example_Schemas/
 http://www.w3.org/2001/XMLSchema" xmlns="urn:hl7-org:v3" xmlns:xsi="http://www.w3.org/2001/
3 <creationTime value="2007080215381000000000" />
4 <securityText>
5 <reference />
6 </securityText>
7 <processingCode code="P" />
8 <processingModeCode />
9 <acceptAckCode code="AL" />
10 <sequenceNumber />
11 <receiver>
12 <typeId assigningAuthorityName="Therapies" />
13 </receiver>
14 <sender>
15 <typeId assigningAuthorityName="RHAPSODY" />
16 </sender>
17 <controlActProcess>
18 <effectiveTime>
19 <high value="2007080215381000000000" />
20 </effectiveTime>
21 <authorOrPerformer>
22 <time>
23 <low />
24 <high value="2007080215381000000000" />
25 </time>
26 </authorOrPerformer>
```

# Kapitel 22

---

MS OOXML Excel-Dateien (und höhere Versionen)

## 22 MS OOXML Excel-Dateien (und höhere Versionen)

Altova Website:  [Mappen von/auf Excel-Dateien](#)

MapForce bietet nun Unterstützung für das Mappen von Microsoft Excel 2007 (und höher) (Office Open XML)-Dateien (\*.xlsx) sowohl als Quell- als auch Zielkomponente.

Microsoft Excel 2007 (oder höher) muss nur dann installiert sein, wenn Sie eine Vorschau der Excel 2007 (und höher)-Arbeitsblätter auf dem Register **Ausgabe** anzeigen wollen. Wenn Sie eine frühere Microsoft Excel-Version verwenden oder Excel gar nicht installiert haben, können Sie trotzdem von/auf Excel 2007 (und höher)-Dateien mappen. Sie können zwar das Ergebnis auf dem Register "Ausgabe" nicht anzeigen, können es aber speichern, indem Sie auf die Schaltfläche "Ausgabe speichern" klicken.

Das Microsoft **Kompatibilitätspaket** für Word, Excel und Powerpoint 2007 (oder höher) genügt nicht, um eine Vorschau der Excel-Ausgabe auf dem MapForce-Register "Ausgabe" anzuzeigen. Sie müssen Excel 2007 oder höher installieren.

Beachten Sie beim Generieren von Code für C#, dass unter **Extras | Optionen | Generierung | C#-Einstellungen** Visual Studio 2005, 2008 oder 2010 ausgewählt sein muss, da nur das .NET Framework 3.0 oder höher das Paketdateiformat von OOXML unterstützt. Die Codegenerierung für Excel 2007 (oder höher) unterstützt Java und C# für das Lesen und Schreiben und XSLT2 nur für das Lesen.

Es gibt zwei Methoden, um gemappte Daten zu generieren/speichern:

- Durch Klicken auf die Schaltfläche "Ausgabe" (oder auf die Schaltfläche BUILTIN). Hierdurch wird mittels des integrierten Built-in-Ausführungsprozessors eine Vorschau in MS Excel 2007 (und höher) generiert. Oder Sie wählen die Menüoption **Ausgabe | Generierte Ausgabe speichern** oder Sie klicken auf die Schaltfläche , um das Ergebnis zu speichern.
- Durch Auswahl von **Datei | Code generieren in | Java, C#** und anschließendes Kompilieren und Ausführen des generierten Codes (Bei der C++-Codegenerierung werden Excel-Dateien nicht unterstützt).

Excel 2007 (und höher)-Dateien können folgendermaßen eingefügt werden:

- als XLSX-Vorlage zusammen mit einer XLSX-Datei, die die Daten für die Mapping-Vorschau liefert (als Quellkomponente).
- als leere XLSX-Vorlage ohne eine XLSX-Datei (als Zielkomponente).

In den folgenden Abschnitten wird beschrieben:

- wie die [mapbaren Datenelemente](#) einer Excel-Datei definiert werden
- wie man mit Excel [Namensbereichen](#) arbeitet
- wie man Zeilenbereiche [hinzufügen, definieren](#) und verschieben kann
- das Mappen von Excel-2007Dateien (und höher) auf [XML](#)
- das Mappen von [Datenbankdaten](#) auf eine Excel 2007 (und höher)-Datei

Beachten Sie bitte, dass es nun möglich ist, Excel 2007 (und höher) Dateien auf eine XBRL-Taxonomiedatei zu mappen. Nähere Informationen dazu finden Sie unter [Beispiel Excel - XBRL](#).



## 22.1 Definieren der mapbaren Datenelemente einer Excel Arbeitsmappe

MapForce kann Tabellendaten in Excel-Arbeitsmappen lesen und schreiben. Die grundlegende hierarchische Struktur einer Excel-Arbeitsmappe ist:

```
Arbeitsmappe
├── Arbeitsblätter
│ └── Zeilen
│ └── Zellen (Schnittmenge aus Zeilen und Spalten)
```

Excel-Arbeitsmappen haben keine Schemas wie XML-Dateien, daher wird ihre Struktur direkt in der MapForce-Komponente definiert. Es können Namen für das Arbeitsblatt und die Spalten definiert werden.

### So fügen Sie eine \*.xlsx-Datei ein:

1. Klicken Sie in der Symbolleiste auf die Schaltfläche "Excel 2007 (und höher)-Datei

einfügen" .

Sie werden nun aufgefordert, eine XLSX-Beispieldatei anzugeben, die die Daten für das Vorschauregister bereitstellt. Anhand der Daten in dieser Datei kann die Baumstruktur der MapForce-Komponente angepasst werden.

2. Klicken Sie auf **Durchsuchen** und wählen Sie die Datei aus, die Sie als Quellkomponente verwenden wollen, z.B. **Nanonull.xlsx**.



Sowohl für die Quell- als auch für die Zielkomponente können manuell Zeilenbereiche definiert werden.

Für Excel-Komponenten können Zeilenbereiche auch über Excel Namensbereiche und Excel-Tabellen aus einer XSL-Datei definiert werden.

Wenn Sie auf eines der Zeilensymbole  klicken, wird ein Dialogfeld geöffnet, in dem Sie den Bereich auswählen/definieren können. Die Arbeitsblätter werden automatisch aus der XLSX-Datei importiert und standardmäßig angezeigt. Sie können dies jedoch ändern.

### Definieren der "mapbaren" Zellen:

Standardmäßig werden alle Zellen einzelner Zeilen in einem Arbeitsblatt durch ein einziges

Datenelement "Zellen" in MapForce angezeigt. Dies bedeutet, dass alle Zellen in den einzelnen Zeilen der Reihe nach verarbeitet werden. Dieser Modus eignet sich dann am besten, wenn alle Spalten in einem Arbeitsblatt dieselbe Bedeutung haben, d.h. wenn jede Zeile eine Liste von Werten ist.

Wenn Sie eine typische Tabellenstruktur mit unterschiedlichen Spalten haben, können Sie stattdessen für jede einzelne Spalte in der MapForce-Struktur separate Datenelemente anzeigen lassen, sodass Sie bestimmte Zellen einfacher aufrufen können:

1. Klicken Sie auf das untere **Zeilen 1, n=dyn** Symbol des Arbeitsblattes "Staff". Daraufhin wird das unten gezeigte Dialogfeld geöffnet.

2. Klicken Sie auf das Optionsfeld "Separate Datenelemente für Spalten anzeigen" und geben Sie die Bezeichnung für das letzte Feld ein, z.B. E.
3. Aktivieren Sie das Kontrollkästchen "Erste Zeile ist Kopfzeile mit den Spaltennamen", um die erste Zeile als Spaltennamen zu verwenden. Die Spalte "Spaltennamen" wird mit den Feldnamen aus dem Excel-Blatt befüllt.

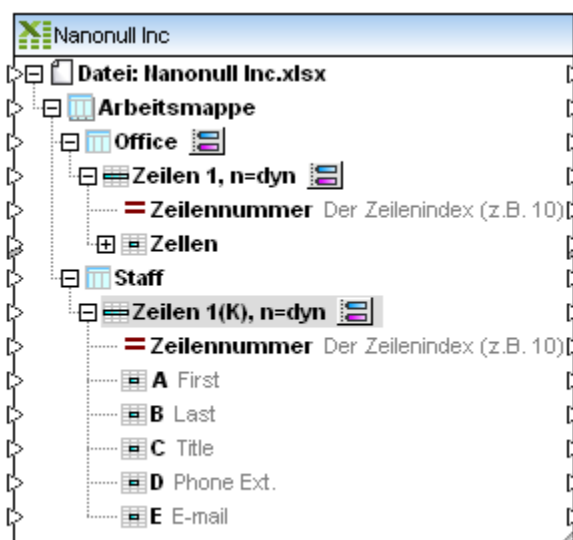
☒ Erste Zeile ist Kopfzeile mit den Spaltennamen Neu laden

Wählen Sie die Zelltypen aus und geben Sie Spaltennamen ein:

|   | Typ    | Spaltenname |
|---|--------|-------------|
| A | string | First       |
| B | string | Last        |
| C | string | Title       |
| D | string | Phone Ext.  |
| E | string | E-mail      |

Sie können jedes der (Daten)-Typ-Felder über die Auswahlliste "Type" ändern.

- Bestätigen Sie die Auswahl mit OK. MapForce zeigt nun unterhalb von Staff | Rows für jede Spalte separate Einträge an.



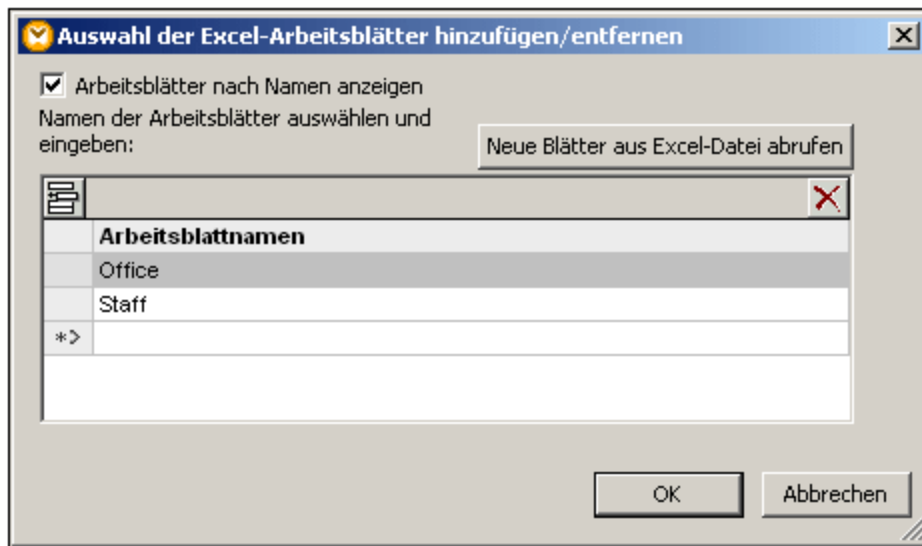
### So definieren Sie die mapbaren Zellen neu (öffnen Sie das Dialogfeld erneut):

- Klicken Sie erneut auf das **Zeilen 1, n=dyn**  Symbol.

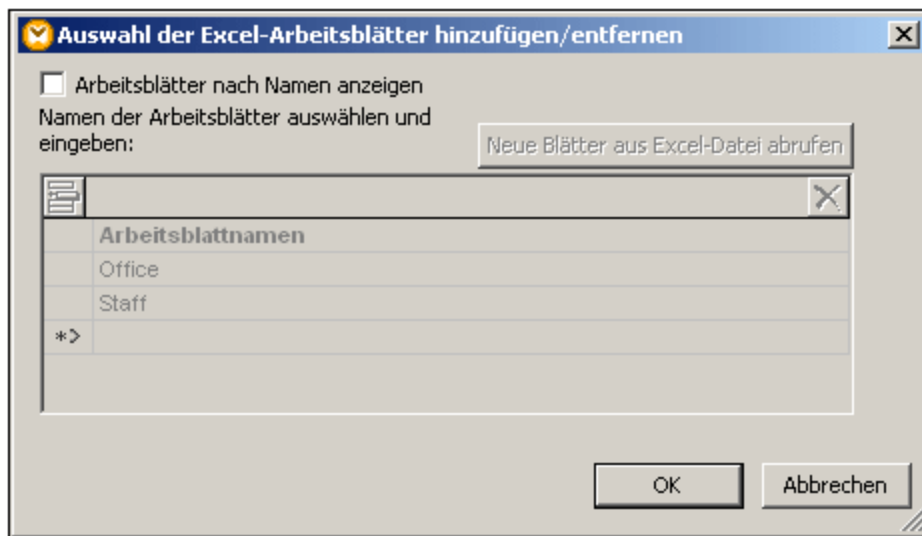
### Definieren der mapbaren Arbeitsblätter:

Normalerweise kann jedes Arbeitsblatt in einer Arbeitsmappe ein anderes Layout haben und wird daher als separater Eintrag in MapForce angezeigt. Wenn alle oder die meisten der Arbeitsblätter in Ihrer Excel 2007 (und höher) Arbeitsmappe eine identische Struktur haben, können Sie die Elemente des Arbeitsblatts zu einem einzigen Datenelement reduzieren, das die geordnete Sammlung aller Arbeitsblätter darstellt:

- Klicken Sie in der Excel-Komponente auf die Schaltfläche  neben dem Eintrag **Office**.  
Die derzeit in der Excel-Datei definierten Arbeitsblätter werden automatisch unter der Überschrift **Arbeitsblattnamen** angezeigt.



2. Klicken Sie auf das Kontrollkästchen "Arbeitsblätter nach Namen anzeigen", um die Auswahl dafür **aufzuheben**.



3. Klicken Sie auf OK. Die Arbeitsmappe wird nun als ein einziger Subeintrag angezeigt, der für alle Arbeitsblätter steht.



Die Struktur der Arbeitsmappe enthält den Arbeitsblattnamen, Zeilen 1, n=dyn und Zelleinträge. Das Datenelemente Zeilen 1, n=dyn steht in diesem Fall für das gesamte Arbeitsblatt.

Bitte beachten Sie:

Wenn eine Arbeitsmappe mehrere Arbeitsblätter enthält, Sie die Option "Arbeitsblätter nach Namen anzeigen" aber **nicht** aktivieren, wird die gesamte Arbeitsmappe als einziges Arbeitsblatt behandelt! Dadurch kann bei einem Mapping eine beliebige Anzahl an Arbeitsblättern auf einmal verarbeitet werden, vorausgesetzt alle diese Arbeitsblätter weisen die gleiche Struktur auf.

### Hinzufügen / Löschen von Arbeitsblättern



Fügt vor dem ausgewählten Arbeitsblatt ein neues ein.



Hängt ein neues Arbeitsblatt an. Geben Sie in das Textfeld rechts vom Symbol einen Namen ein.



Löscht das aktuell ausgewählte Arbeitsblatt.

Klicken Sie auf die Schaltfläche "Neue Blätter aus Excel-Datei abrufen" um neue in der XSLX-Eingabedatei gefundene Blätter anzuhängen.

### Excel 2007 (und höher)-Komponenteneinstellungen

Die Input und Output-Instanzdateien können hier mit der Option eine Excel 2007 (und höher)-Datei als globale Ressource auszuwählen, ausgewählt werden. Nähere Informationen dazu finden Sie unter [Globale Ressourcen](#).

Die Anzeige von Fehlermeldungen, wenn der vom Benutzer ausgewählte Spaltendatentyp nicht mit den Daten in der XSLX-Datei übereinstimmt, im Fenster "Meldungen" kann für Excel-Quellkomponenten durch Aktivierung des Optionsfelds "Fehler bei nicht übereinstimmendem Typ anzeigen" aktiviert/deaktiviert werden.

Wenn die Spalte z.B. als numerisch definiert ist, wird bei eingeschalteter Option "Fehler bei nicht übereinstimmendem Typ anzeigen", eine Fehlermeldung bzw. ein Ausnahmeereignis angezeigt, wenn die XSLX-Datei den Text "unknown" enthält.

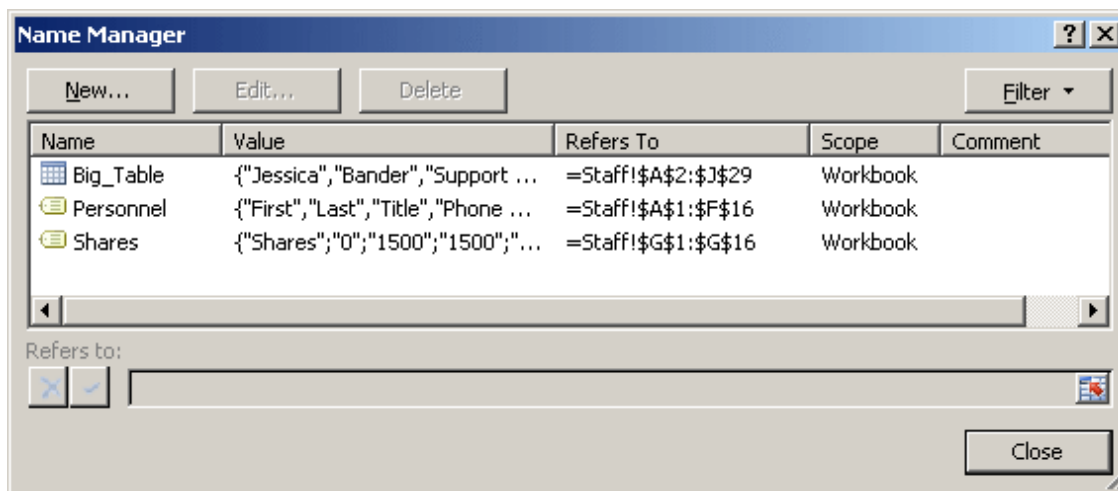


## 22.2 Verwenden von Excel-Namen für Bereiche und Excel-Tabellen

MapForce ist in der Lage:

- Excel-Namen für Bereiche und Tabellen zu lesen und die Daten als mapbare Datenelemente in einer Quellkomponente darzustellen.

Nachdem Sie in der Arbeitsmappe Nanonull.xlsx Bereiche und Tabellen definiert haben, öffnen Sie den Namens-Manager in Excel, um diese anzuzeigen. In der Abbildung unten sehen Sie, dass im Arbeitsblatt "Staff" eine Tabellen und zwei Namensbereiche definiert wurden.



Anmerkung: Derzeit werden in MapForce nur rechteckige Bereiche unterstützt.

### Auswahl eines Namensbereichs aus einem Excel-Arbeitsblatt:

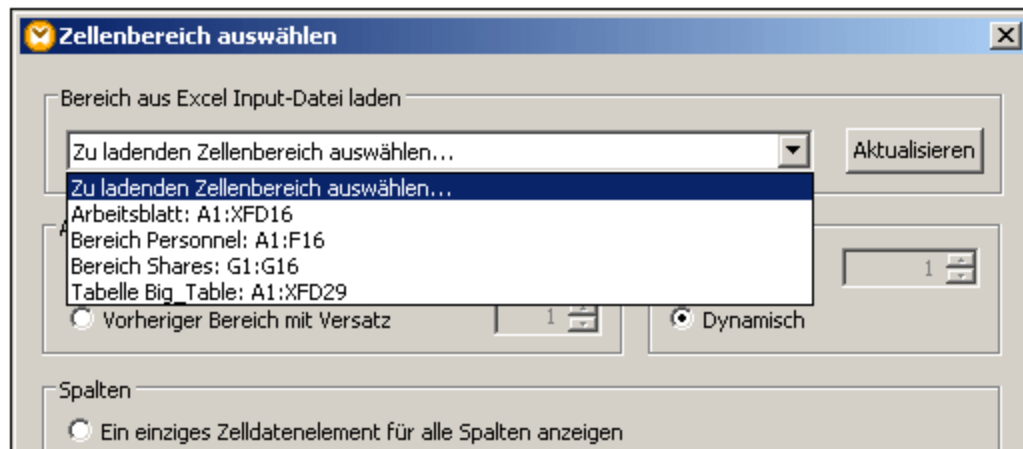
Öffnen Sie zuerst die Datei **Excel\_Company\_to\_XML.mfd** aus dem Verzeichnis ... \MapForceExamples:

1. Klicken Sie auf das **Zeilen 1(K), n=dyn** Symbol unterhalb des Staff-Arbeitsblattsymbols.



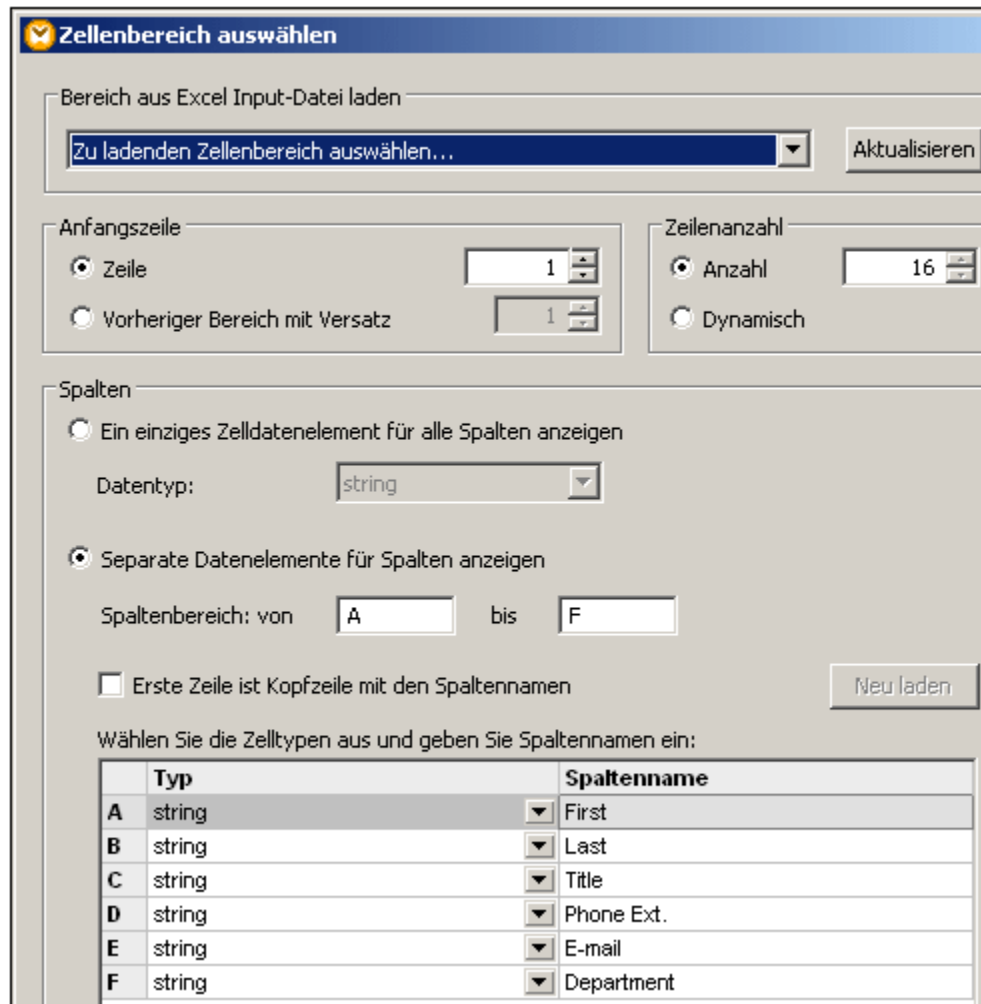
Daraufhin wird das Dialogfeld "Zellenbereich auswählen" geöffnet, in dem Sie die Zellenbereiche der Excel-Komponente definieren.

2. Klicken Sie auf die Auswahlliste, um die derzeit definierte Bereiche, Tabellen und die Größe des Arbeitsblattes (Zellbezüge) zu sehen.



Die Namen der Bereiche/Tabellen sowie die Zellbezüge werden alle in der Dropdown-Auswahlliste angezeigt.

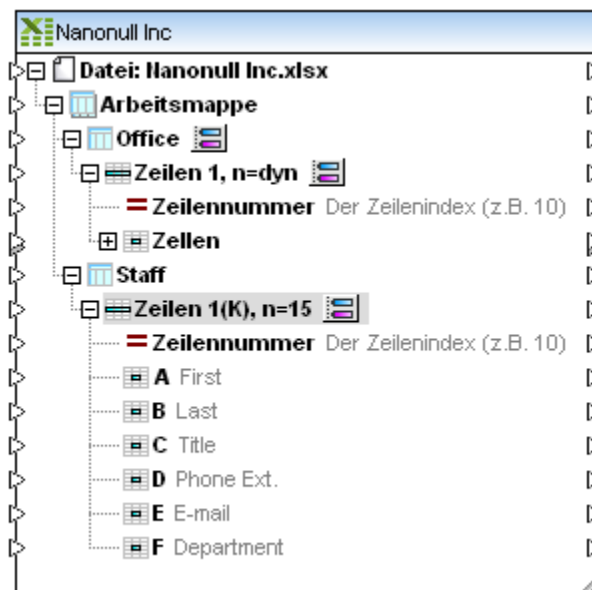
3. Bei Auswahl des Bereichs "Personnel: A1:F16" werden im Dialogfeld automatisch die Informationen/Koordinaten zum Namensbereich angezeigt.



- Die Anfangszeile ist 1



- Die Zeilenanzahl beträgt 16
  - Der Spaltenbereich erstreckt sich von A bis F
  - Das Optionsfeld "Separate Datenelemente für Spalten anzeigen" ist aktiv.
4. Aktivieren Sie das Kontrollkästchen "Erste Zeile ist Kopfzeile" um die Spaltennamen als Kopfzeile zu verwenden.
  5. Klicken Sie im anschließend angezeigten Meldungsfeld auf "Ja", um die Zeilenanzahl zu verringern.
- Beachten Sie, dass die Kopfzeile nicht in der Zeilenanzahl enthalten ist. Der Excel Namensbereich "Personnel" besteht aus einer Kopfzeile und 15 Datenzeilen.
6. Klicken Sie auf OK, um das Dialogfeld zu schließen.



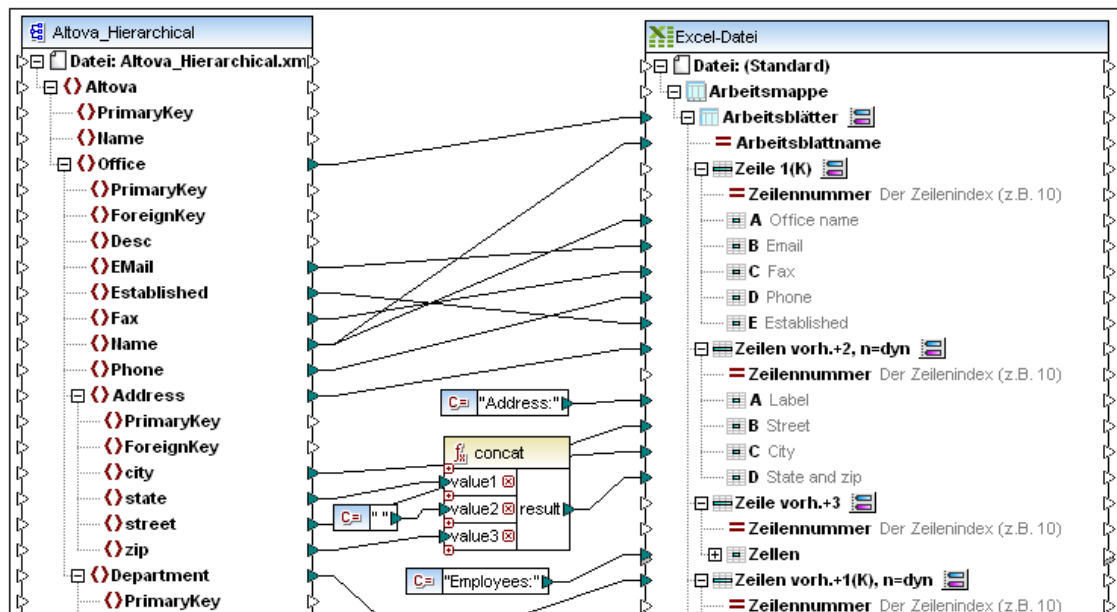
Die Datenelemente des Arbeitsblattes "Staff" haben sich jetzt geändert. Neben dem Datenelement "Zeilen" wird jetzt eine kurze Info zum Bereich angezeigt.

**Zeilen 1(K), n=15**, bedeutet dasselbe wie oben gezeigt: Beginn bei Zeile 1, Kopfzeile aktiv, die Zeilenanzahl beträgt 15.

## 22.3 Hinzufügen, Definieren und Verschieben von Zeilenbereichen

Sowohl für die Excel-Quell- als auch die Excel- [Excel-Zielkomponente](#) können Zeilenbereiche definiert werden. Im Allgemeinen wird diese Funktion jedoch für die Definition der Zielkomponente verwendet.

In diesem Abschnitt wird erläutert, wie das Mapping **Altova\_Hierarchical\_Excel.mfd** erstellt wurde. Sie finden das Beispiel im Ordner ...\\MapForceExamples.



Wenn Sie ein Excel-Arbeitsblatt als Zielkomponente einfügen (d.h. es steht keine XLSX-Beispieldatei zur Verfügung), wird automatisch eine Arbeitsmappe mit drei Arbeitsblättern vorgegeben.

Wenn Sie nur mit Arbeitsblättern arbeiten möchten, wie im Beispiel unten, gehen Sie folgendermaßen vor:

- Klicken Sie auf das **Blatt1**  Symbol und deaktivieren Sie im Dialogfeld das Kontrollkästchen "Arbeitsblätter nach Namen anzeigen".

Arbeitsblätter aktiv



Arbeitsblätter deaktiviert



### So erstellen Sie den ersten Bereich bzw. fügen diesen hinzu:

1. Klicken Sie mit der rechten Maustaste auf das erste Zeilen-Datenelement (**Zeilen 1, n=dyn**) um das Dialogfeld "Zellenbereich auswählen" zu öffnen.

2. Hier wurde Anfangszeile =1 und Anzahl = 1 ausgewählt, da der Name des Büros nur eine Zeile im Excel-Arbeitsblatt beanspruchen sollte.

Beachten Sie, dass die XML-Quellinstanz Daten zu zwei Nanunulll-Büros, nämlich Nanonull Inc und Nanonull Partners, Inc, enthält. Diese Excle-Komponente generiert daher zwei Arbeitsblätter in der Excel-Arbeitsmappe. Die Daten für die Arbeitsblattnamen stammen aus dem Node **Office | Name** der XML-Quellinstanz.

Die **Spaltennamen** der gemappten Office-Datenelemente werden folgendermaßen definiert:

- Aktivieren Sie das Optionsfeld "Separate Datenelemente für Spalten anzeigen" und geben Sie das letzte Spaltenelement an, z.B. "E".
- Anschließend werden die Spalten in Listenfeld unterhalb generiert. Alle Spalten haben den Typ "string".
- Anschließend werden die Spaltennamen wie unten gezeigt direkt in die Spalte "Spaltenname" des Listenfelds eingegeben. (Über die Auswahlliste "Typ" können Sie auch den Zelltyp, z.B. "date" ändern.)

Spalten

☐ Ein einziges Zelldatenelement für alle Spalten anzeigen

Datentyp:

☒ Separate Datenelemente für Spalten anzeigen

Spaltenbereich: von  bis

☒ Erste Zeile ist Kopfzeile mit den Spaltennamen Neu laden

Wählen Sie die Zelltypen aus und geben Sie Spaltennamen ein:

|   | Typ    | Spaltenname |
|---|--------|-------------|
| A | string | Office name |
| B | string | Email       |
| C | string | Fax         |
| D | string | Phone       |
| E | Datum  | Established |

Neuer Bereich: **Zeile1 (K)**

#### Hinzufügen eines neuen Bereichs:

1. Klicken Sie mit der rechten Maustaste auf das erste Zeilen-Datenelement (**Zeilen 1, n=dyn**) und wählen Sie im Kontextmenü den Befehl "Zeilen danach hinzufügen ...".

Bereich aus Excel Input-Datei laden

Aktualisieren

Anfangszeile

☐ Zeile

☒ Vorheriger Bereich mit Versatz

Zeilenanzahl

☐ Anzahl

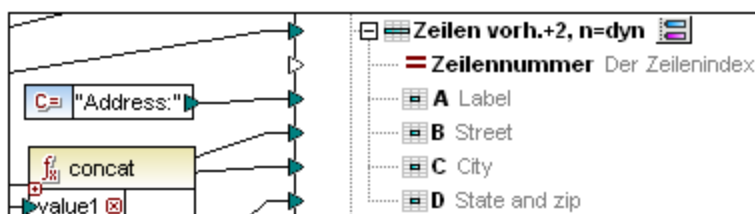
☒ Dynamisch

Der nächste Bereich beginnt mit einem Versatz 2 nach dem vorherigen Bereich, d.h. es wird eine leere Zeile zwischen den beiden Bereichen gelassen und die Zeilenanzahl ist "dynamisch".

Auf dieselbe oben beschriebene Art werden die Spaltennamen für die Spalten A bis D erstellt.

Dieser Bereich hat keine Kopfzeile, daher bleibt das Kontrollkästchen "Erste Zeile ist Kopfzeile mit den Spaltennamen" deaktiviert.

Beachten Sie, dass die Daten für den ersten Spaltennamen (Label) aus der Konstantenkomponente stammen. Die Konstantenkomponente enthält den String "Address".



Neuer Bereich: **Zeilen vorh.+2, n=dyn**

### Hinzufügen der darauffolgenden Bereiche:

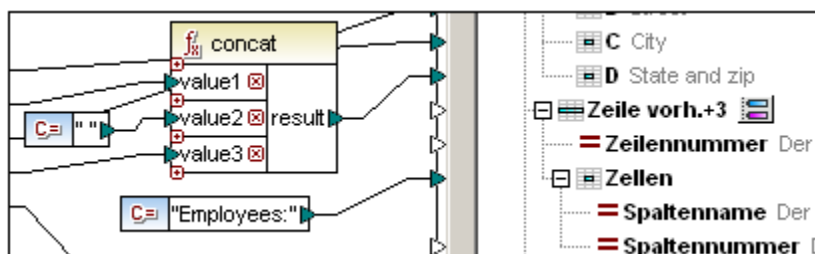
Gehen Sie wie oben beschrieben vor, um die neuen Bereiche zur Excel-Komponente hinzuzufügen.

### Zeile vorh.+3

Dieser nächste Bereich beginnt mit einem Versatz von 3 nach dem vorherigen Bereich und hat eine Zeilenanzahl 1.

Beachten Sie, dass das Optionsfeld "Ein einziges Zelldatenelement für alle Spalten anzeigen" hier **aktiv** ist. Es sind keine Spaltennamen definiert.

Mit dieser Bereichsdefinition werden zwei leere Zeilen eingefügt und eine Zeile wird eingefügt, die mit Daten befüllt wird. Die Daten für die dritte Zeile stammen aus einer Konstantenkomponente mit dem String "Employees" und werden auf das Datenelement "Zellen" in der Zielkomponente gemappt.



**Zeilen vorh.+1(K), n=dyn**

Der nächste Bereich beginnt mit einem Versatz von 1 nach dem vorherigen Bereich. Die erste Zeile wird als Kopfzeile verwendet, wobei die Zeilenanzahl dynamisch ist.

Das bedeutet, dass die Anzahl der in der Excel-Zielkomponente erstellten Zeilen dynamisch ist und von den von der XML-Quellkomponente gelieferten Personendaten abhängt.

Die Spaltennamen für die Spalten A bis E werden auf die oben beschriebene Art erstellt.

### Zeile vorh.+3

Dieser Bereich wird wiederholt und fügt zwei leere Zeilen ein, wobei die dritte die von der Konstantenkomponente gelieferte "List of departments" (Abteilungsliste) enthält.

### Zeilen vorh.+1, n=dyn

|   | Typ    | Spaltenname |
|---|--------|-------------|
| B | string | Name        |

Dieser Bereich beginnt mit einem Versatz von 1 nach dem vorherigen Bereich und ist dynamisch.

Beachten Sie, dass der Spaltenbereich als B bis B definiert ist, d.h. es gibt nur eine Spalte und die Daten dafür stammen aus dem Node **Department | Name** der XML-Quellinstanz.

Die einzelnen Abteilungsnamen werden in die Spalte B eingefügt.

### So fügen Sie einen Bereichseintrag in eine Excel-Komponente ein:

- Klicken Sie mit der rechten Maustaste auf einen vorhandenen Bereichseintrag (z.B. Zeilen 2, n=dyn) und wählen Sie, je nachdem, wo der neue Bereich im Arbeitsblatt

eingefügt werden soll, entweder den Befehl "Zeilen davor hinzufügen..." oder "Zeilen danach hinzufügen..."

**So verschieben Sie einen Bereich in einer Excel-Komponente:**

- Klicken Sie mit der rechten Maustaste auf den gewünschten Bereich und wählen Sie im Kontextmenü den Befehl "Nach oben" oder "Nach unten".

**So entfernen Sie einen Bereich aus einer Excel-Komponente:**

- Klicken Sie mit der rechten Maustaste auf den gewünschten Bereichseintrag und wählen Sie im Kontextmenü den Befehl "Zeilen entfernen".

## 22.4 Zellenbereich auswählen - Optionen

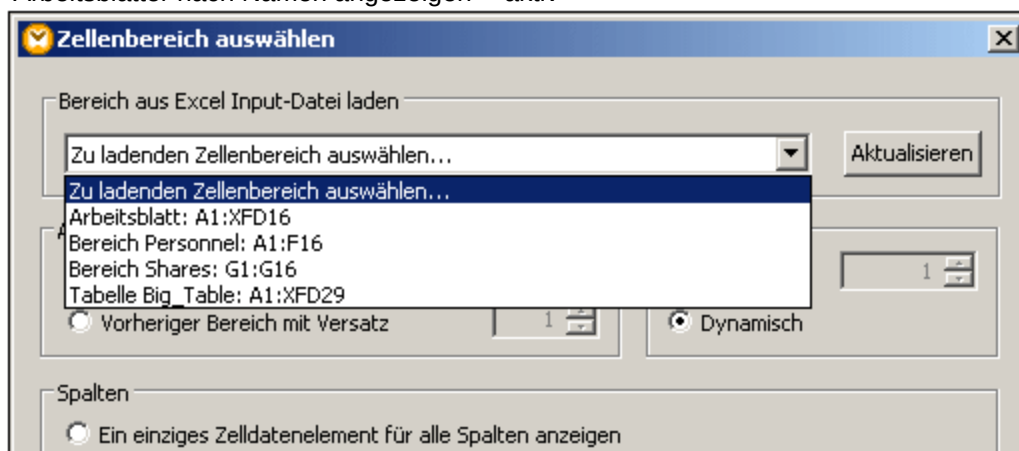
### Bereich aus Excel Input-Datei laden

Mit Hilfe dieser Option können Sie Excel-Namen für Bereiche, Excel-Tabellen (oder das gesamte Arbeitsblatt) auswählen, falls diese zuvor in der Excel-Datei definiert wurden. Die Bereichs-/Tabellennamen sowie die Zellbezüge werden alle in der Drop-Down-Auswahlliste angezeigt.

Beachten Sie, dass derzeit nur rechteckige Bereiche von MapForce unterstützt werden.

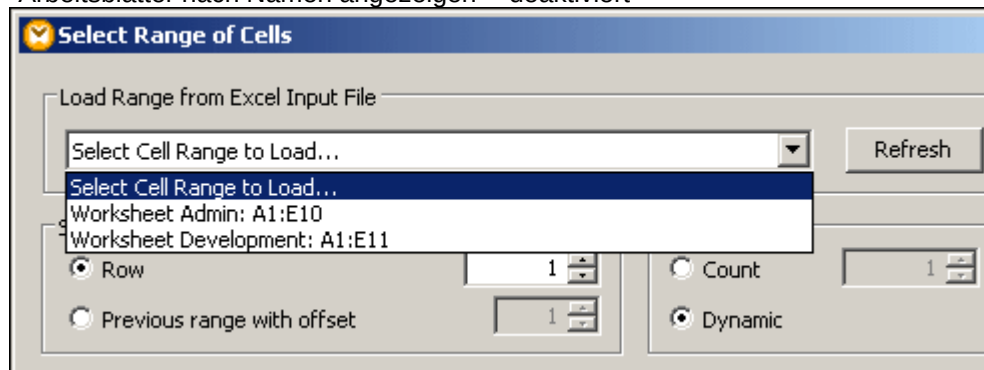
Nachdem Sie einen Bereich ausgewählt und mit OK bestätigt haben, wird der neue Bereich in der Komponente angezeigt. Wenn Sie das Dialogfeld erneut öffnen, wird statt dem zuvor ausgewählten Bereich der Eintrag "Zu ladenden Zellenbereich auswählen" angezeigt.

"Arbeitsblätter nach Namen anzeigen" - aktiv



Wenn Sie auf die Schaltfläche "Aktualisieren" klicken, wird die Excel-Datei neu eingelesen und die Bereiche werden aktualisiert, falls sie sich geändert haben.

"Arbeitsblätter nach Namen anzeigen" - deaktiviert



In der obigen Abbildung sehen Sie die verfügbaren Bereiche, wenn die Option "Arbeitsblätter nach Namen anzeigen" deaktiviert ist. In diesem Fall werden die einzelnen Arbeitsblätter nicht in der Excel-Komponente angezeigt. Die in der Auswahlliste "Bereich aus Excel Input-Datei laden" verfügbaren Bereiche sind die in der XLSX-Datei definierten einzelnen Arbeitsblätter (Admin and Development).

**Anfangszeile: Zeile**



Mit Hilfe dieser Option können Sie die erste Datenzeile für den jeweiligen Bereich definieren. Wenn Sie z.B. für den ersten Bereich 5 eingeben, so wird als erste Zeile aus dem Blatt die Zeile 5 verwendet.

Wenn Sie auch Anzahl auf 1 setzen, so erhält der nächste Bereich, den Sie definieren, als Anfangszeile die Zeile 6, wenn die Option "Vorheriger Bereich mit Versatz" aktiv ist. Ein neuer Bereich kann natürlich auch eine absolute Bereichsdefinition haben, z.B. Anfangszeile=25 und Zeilenanzahl=5.

Wenn Sie zusätzlich das Kontrollkästchen "Erste Zeile enthält Spaltennamen" aktivieren, werden die Felder der Zeile 5 als Spaltenüberschriften verwendet.

### **Anfangszeile: Vorheriger Bereich mit Versatz**

Mit dieser Option können Sie die Anfangszeile des nächsten Bereichs automatisch definieren, d.h. die letzte Zeile des vorherigen Bereichs wird berücksichtigt.

Der Versatz definiert, wie viele Extrazeilen zum automatisch ermittelten Beginn des Bereichs hinzugefügt werden sollen. Der Mindestversatz ist 1.

### **Zeilenanzahl: Anzahl**

Wenn Bereiche in einer Excel-**Quelldatei** definiert werden, so ist "Anzahl" entweder:

- die manuell eingegebene Anzahl der Zeilen, die zur Anfangszeile hinzugefügt werden sollen.
- die automatisch ermittelte Anzahl der Zeilen, wenn Sie einen Excel-Namensbereich oder eine Excel-Tabelle ausgewählt haben.

Anmerkung: In der Anzahl ist die Kopfzeile nicht berücksichtigt!

### **Zeilenanzahl: Dynamisch**

Wenn Bereiche in einer Excel-**Zieldatei** definiert werden, so bedeutet "dynamisch", dass die Anzahl der Zellen von den Quelldaten, die auf den Bereich gemappt wurden, abhängig ist.

Wenn in einer Excel-**Quelldatei** ein Bereich definiert wird, bedeutet "dynamisch", dass der Bereich prinzipiell unbegrenzt ist. Anmerkung: Alle **darauf folgenden** Bereiche, die in diesem Fall definiert sind, wählen niemals Daten aus der Excel-Quelldatei aus.

### **Spalten: Ein einziges Zelldatenelement für alle Spalten anzeigen:**

Reduziert alle Zelldatenelemente zu einem einzigen mapbaren Zelldatenelement, wie unten gezeigt. Ein Beispiel dazu finden Sie unter [Hinzufügen, Definieren und Verschieben von Zellenbereichen](#).



**Spalten: Separate Datenelemente für Spalten anzeigen:**

Mit dieser Option werden die zuvor definierten Arbeitsblätter der Excel-Arbeitsmappe angezeigt. Sie können darin die Anzahl der Spalten und deren Namen definieren.



Die Spaltennamen werden automatisch eingefügt, wenn Sie eine Excel-Tabelle auswählen. Die Spaltennamen können auch manuell durch Eingabe der Namen in die Spalte "Spaltenname" definiert werden.

Wenn das Kontrollkästchen "Erste Zeile ist Kopfzeile mit den Spaltennamen" aktivieren, werden die Spaltennamen nur dann automatisch ausgewählt, wenn das Arbeitsblatt und die Anfangszeile ordnungsgemäß definiert wurden.

**Spalten**

☐ Ein einziges Zelldatenelement für alle Spalten anzeigen

Datentyp:

☒ Separate Datenelemente für Spalten anzeigen

Spaltenbereich: von  bis

☒ Erste Zeile ist Kopfzeile mit den Spaltennamen Neu laden

Wählen Sie die Zelltypen aus und geben Sie Spaltennamen ein:

|   | Typ    | Spaltenname |
|---|--------|-------------|
| A | string | First       |
| B | string | Last        |
| C | string | Title       |
| D | string | Phone Ext.  |
| E | string | E-mail      |
| F | string | Department  |

**Erste Zeile ist Kopfzeile mit den Spaltennamen**

Mit dieser Option können Sie definieren, ob die erste Zeile eines Bereichs als Kopfzeile für diesen Bereich verwendet werden soll.

Beachten Sie, dass die Kopfzeile bei der Einstellung Zeile | Anzahl nicht berücksichtigt wird.

Wenn Sie diese Option aktivieren/deaktivieren und die Zeilenanzahl definiert wurde, erscheint eine Bestätigungsaufforderung, in der Sie gefragt werden, ob Sie den Wert der Zeilenanzahl anpassen möchten. Dadurch wird vermieden, dass die Zeilenanzahl eine Zeile zu viel bzw. zu

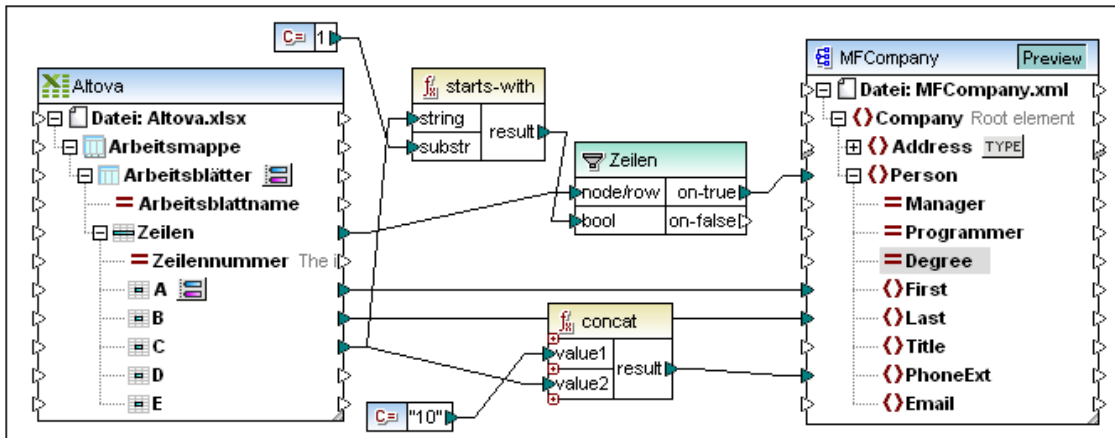
wenig enthält.

## 22.5 Mappen von Excel-Dateien auf XML

Die im folgenden Beispiel verwendete Mapping-Datei steht im Ordner **...MapForceExamples\Tutorial\** unter dem Namen **Excel-mapping.mfd** zur Verfügung. Die beiden Mappings werden in diesem Abschnitt erläutert.

**Ziel** des Mappings ist Folgendes:

- Filtern nach jener Personen im Excel-Arbeitsblatt, deren Durchwahl mit einer "1" beginnt (in Spalte C in der Arbeitsmappe).
- Hinzufügen/Anhängen einer "10" an die ursprüngliche Zahl und Platzieren der Zahl in die XML-Datei - zusammen mit den Vor- und Zunamen der entsprechenden Personen.



- **Altova.xlsx** ist die Quelldatei/-komponente. Die Spalten A und B stellen die Vor- bzw. Zunamen zur Verfügung. Spalte C enthält die Telefondurchwahl.
- Die einzelnen **Arbeitsblätter** der Arbeitsmappe wurden nicht aktiviert (Option "[Arbeitsblätter nach Namen anzeigen](#)"). Sie sehen dies am allgemeinen Titel "Arbeitsblätter" unter dem Eintrag Arbeitsmappe.
- Die Funktion **starts-with** überprüft, ob die Durchwahl (Spalte C) mit einer "1" beginnt. Wenn das Ergebnis "true" ist, werden diese Datensätze von der Filterkomponente übergeben.
- Der concat-Filter stellt den einzelnen Durchwahlen eine "10" voran und übergibt sie an den Eintrag PhoneExt.
- MFCompany.xsd ist die Zielkomponente und enthält bei Ausgabe der Daten die gefilterten Personendaten.

**Ergebnis** des Mappings:

Es wurden vier Personen mit ihren Daten auf die XML-Datei gemappt.

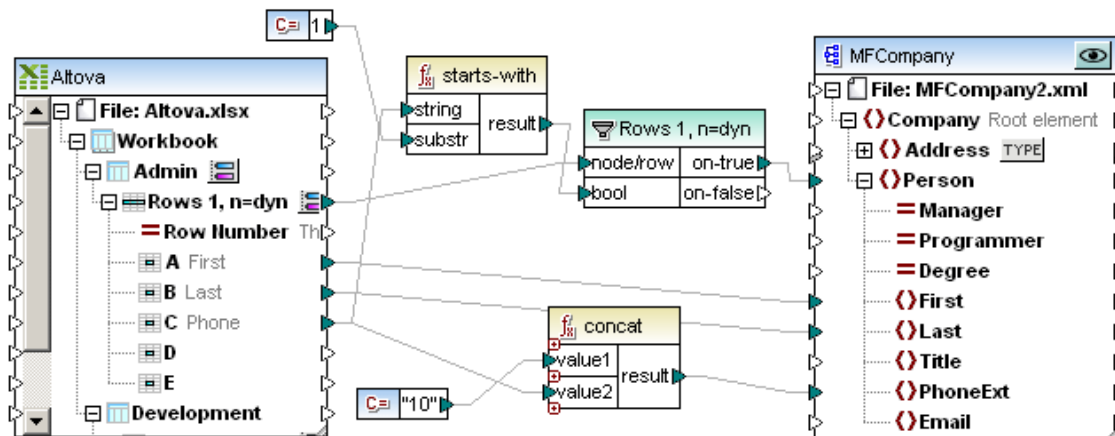
```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <Company xsi:schemaLocation="http://my-company.com/namespace
3 <Person>
4 <First>Steve</First>
5 <Last>Meier</Last>
6 <PhoneExt>10114</PhoneExt>
7 </Person>
8 <Person>
9 <First>Max</First>
10 <Last>Nafta</Last>
11 <PhoneExt>10122</PhoneExt>
12 </Person>
13 <Person>
14 <First>Carl</First>
15 <Last>Franken</Last>
16 <PhoneExt>10147</PhoneExt>
17 </Person>
18 <Person>
19 <First>Mark</First>
20 <Last>Redgreen</Last>
21 <PhoneExt>10152</PhoneExt>
22 </Person>
23 </Company>

```

Im zweiten Mapping wird das Ergebnis angezeigt, wenn die XLSX-Komponenten **Arbeitsblätter** mit Hilfe der Option [Arbeitsblätter nach Namen anzeigen](#) einzeln aktiviert wurden:

- Die Arbeitsblätter **Admin** und **Development** werden beide unterhalb des Eintrags Arbeitsmappe angezeigt.
- Es wurden nur Konnektoren vom Arbeitsblatt **Admin** auf die Zielkomponente definiert.



**Ergebnis** des Mappings:

Es wurden zwei Personen mit ihren Daten auf die XML-Datei gemappt.

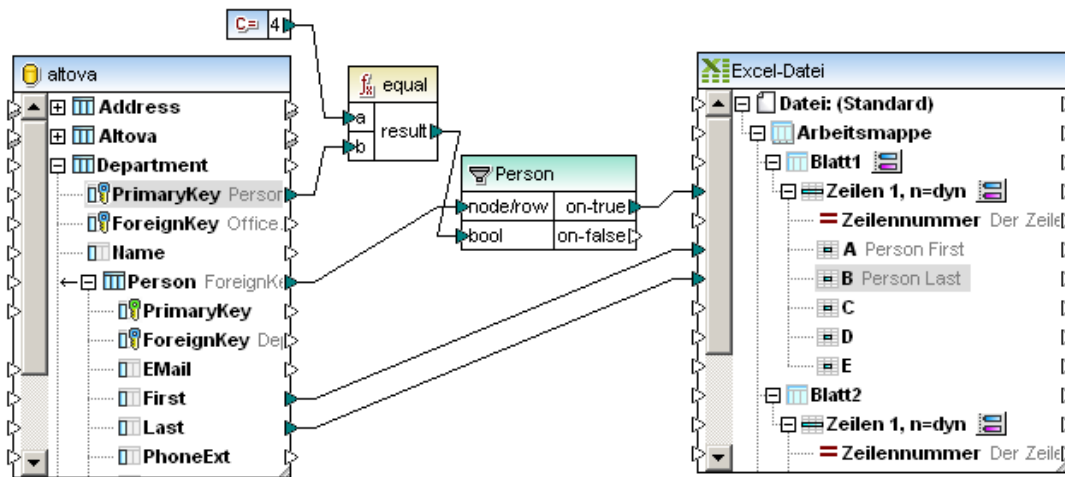
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Company xmlns:xs="http://my-company.com/namespace"
3 <Person>
4 <First>Steve</First>
5 <Last>Meier</Last>
6 <PhoneExt>10114</PhoneExt>
7 </Person>
8 <Person>
9 <First>Max</First>
10 <Last>Nafta</Last>
11 <PhoneExt>10122</PhoneExt>
12 </Person>
13 </Company>
```

## 22.6 Mappen von Datenbankdaten auf Excel

Die im folgenden Beispiel verwendete Mapping-Datei steht im Ordner **...MapForceExamples\Tutorial\** unter dem Namen **Excel-mapping.mfd** zur Verfügung. Das unterste der drei Mappings wird hier erläutert.

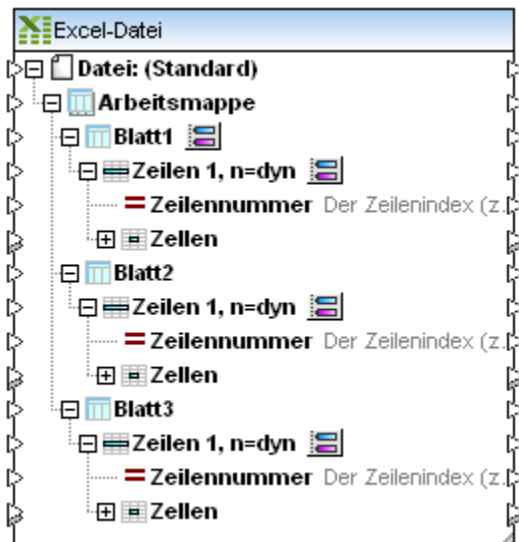
**Ziel** des Mappings ist folgendes:

- Filtern nach jenen Personen in der Altova-Datenbank, deren Primärschlüssel für die Abteilung "4" ist, d.h. die der IT-Abteilung angehören.
- Einfügen der Personen aus der IT-Abteilung in die "leere" Excel-Dateivorlage.



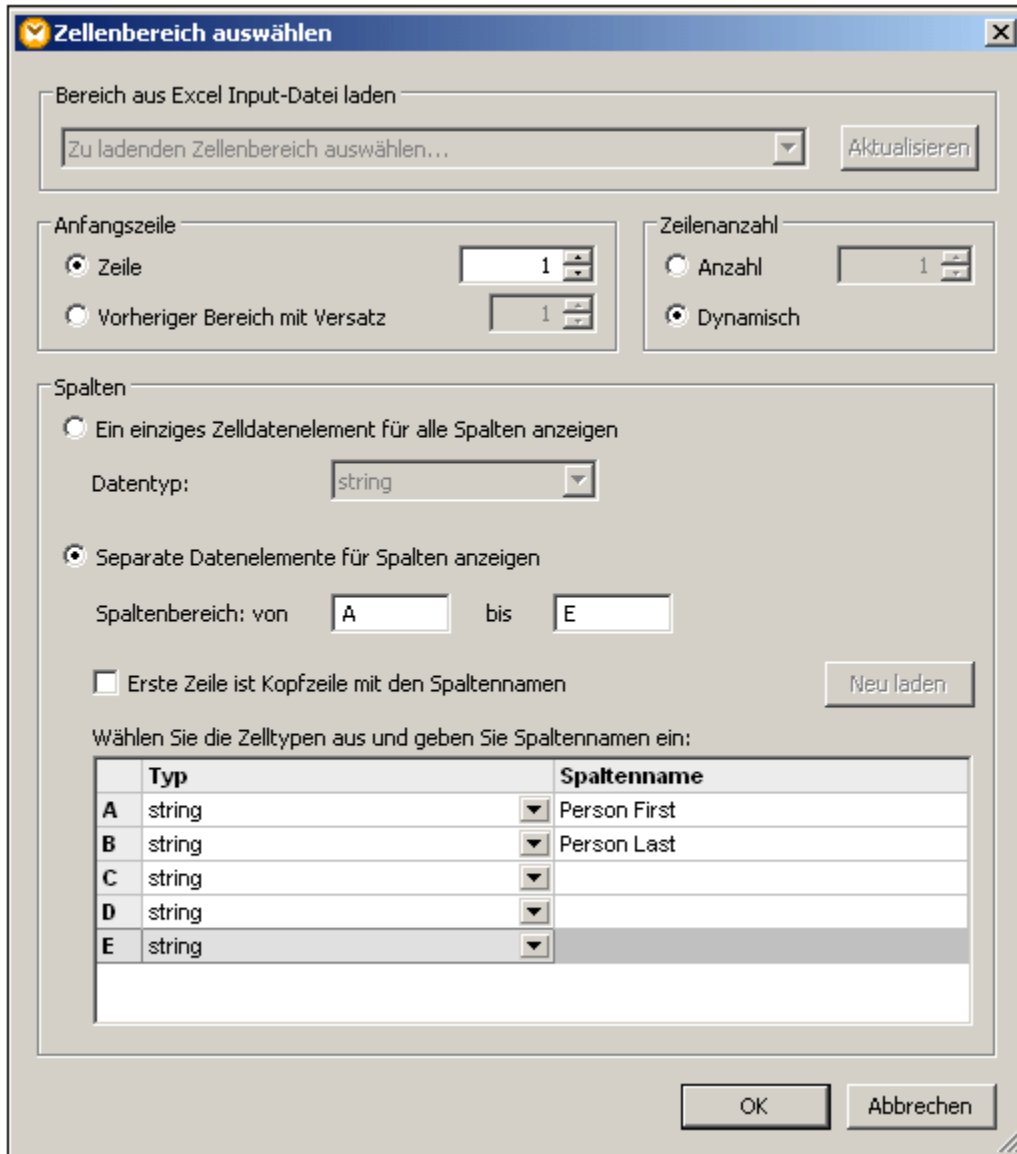
**So erstellen Sie eine leere Excel-Arbeitsmappenvorlage.**

1. Klicken Sie in der Symbolleiste auf die Schaltfläche "Excel 2007-Datei einfügen" . Sie werden nun aufgefordert, eine XLSX-Beispieldatei für die Daten des Vorschauregisters anzugeben.
2. Klicken Sie auf **Überspringen**.



Es wird eine leere Excel-Komponente eingefügt.

3. Klicken Sie (unter Blatt1) auf das Symbol **Zeilen 1, n=dyn** , um festzulegen, wie viele Spalten das Arbeitsblatt enthalten soll.
4. Aktivieren Sie das Optionsfeld "Separate Datenelemente für Spalten anzeigen" und geben Sie in die Textfelder den Zellenbereich A bis E an.



**Zellenbereich auswählen**

Bereich aus Excel Input-Datei laden

Zu ladenden Zellenbereich auswählen... Aktualisieren

Anfangszeile

☒ Zeile 1

☐ Vorheriger Bereich mit Versatz 1

Zeilenanzahl

☐ Anzahl 1

☒ Dynamisch

Spalten

☐ Ein einziges Zelldatenelement für alle Spalten anzeigen

Datentyp: string

☒ Separate Datenelemente für Spalten anzeigen

Spaltenbereich: von A bis E

☐ Erste Zeile ist Kopfzeile mit den Spaltennamen Neu laden

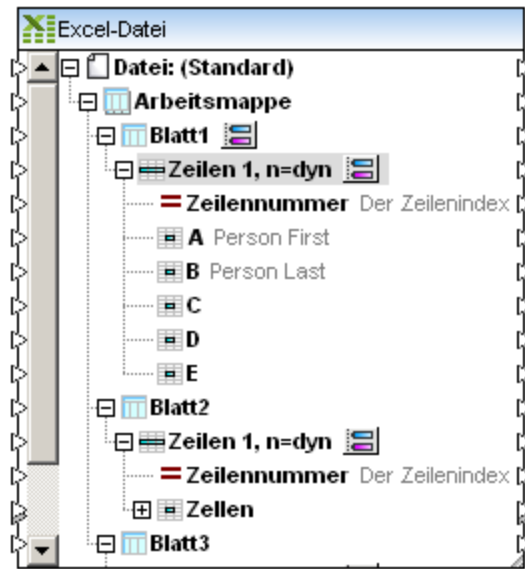
Wählen Sie die Zelltypen aus und geben Sie Spaltennamen ein:

|   | Typ    | Spaltenname  |
|---|--------|--------------|
| A | string | Person First |
| B | string | Person Last  |
| C | string |              |
| D | string |              |
| E | string |              |

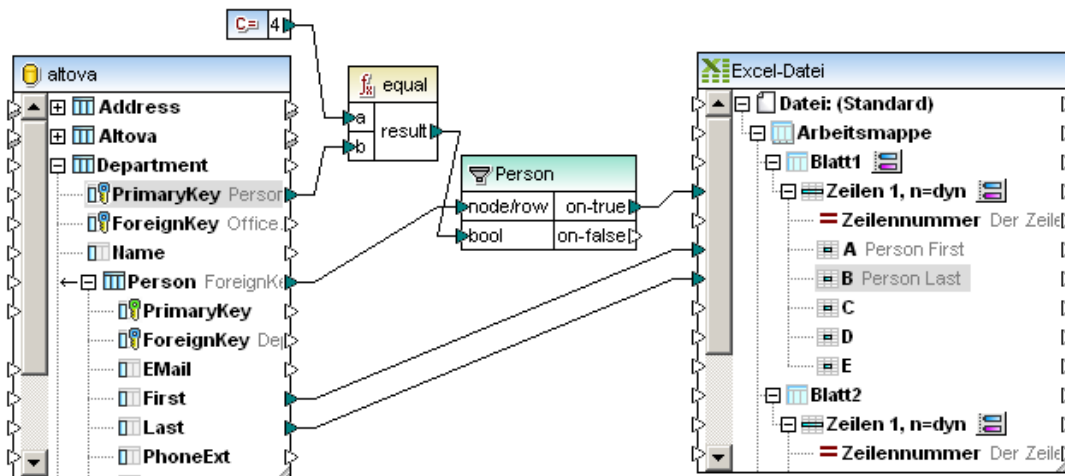
OK Abbrechen

5. Klicken Sie in die Spaltennamenfelder und geben Sie einen beschreibenden Text zum Mapping ein, z.B. Person First/Last und klicken Sie auf OK, wenn Sie fertig sind.





Das Arbeitsblatt enthält nun die Spalten A bis E, auf die Sie die Komponentenkonnektoren mappen können.



- Der Wert von PrimaryKey wird mit Hilfe der equal-Funktion mit dem Wert 4 aus der Konstantenkomponente verglichen.
- Die Filterkomponente übergibt die First- und Last-Felder, bei denen die Boolesche Bedingung "true" ergibt. Der Inhalt der Subeinträge/Nodes, die mit dem node/row-Parameter der Quellkomponente verbunden sind, wird an die Zielkomponente übergeben.
- Beachten Sie, dass das Datenelement on-true-mit dem Datenelement **Zeilen 1, n=dyn** in der Excel-Datei verbunden ist.
- Der Arbeitsblattname "Altova" wurde durch Klicken auf das Sheet1 Symbol und Eingabe von "Altova" als Arbeitsblattname definiert.

#### Ergebnis des Mappings:

Die vier Personen der IT-Abteilung werden in der Excel-Arbeitsmappe angezeigt.

|   | A       | B      | C | D |
|---|---------|--------|---|---|
| 1 | Alex    | Martin |   |   |
| 2 | George  | Hammer |   |   |
| 3 | Jessica | Bander |   |   |
| 4 | Lui     | King   |   |   |
| 5 |         |        |   |   |
| 6 |         |        |   |   |

Altova

# Kapitel 23

---

## Bibliotheken und Funktionen

## 23 Bibliotheken und Funktionen

In den folgenden Abschnitten wird beschrieben, wie Sie Ihre eigenen benutzerdefinierten Funktionen sowie Bibliotheken für die verschiedenen Programmiersprachen definieren.

[Definieren benutzerdefinierter Funktionen](#)

[Hinzufügen benutzerdefinierter XSTL- und XQuery-Funktionen](#)

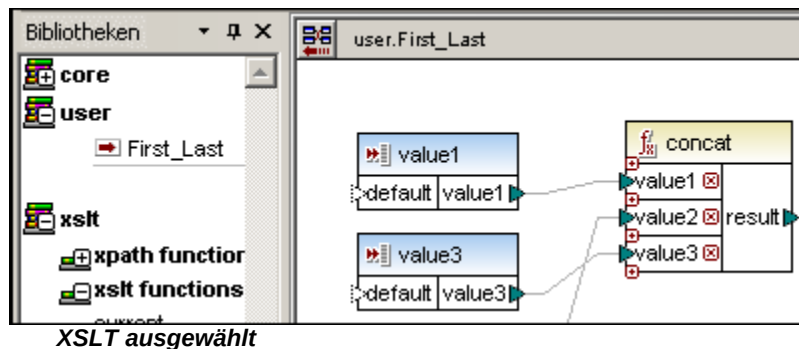
[Hinzufügen benutzerdefinierter Java, C# und C++-Funktionsbibliotheken](#)

[Referenz Funktionsbibliotheken](#)

## 23.1 Definieren benutzerdefinierter Funktionen

In MapForce können Sie benutzerdefinierte Funktionen visuell auf dieselbe Art wie im Hauptmappingfenster definieren.

Diese Funktionen stehen dann im Fenster "Bibliotheken" als Einträge zur Verfügung (z.B. First\_Last) und können wie die bereits vorhandenen Funktionen verwendet werden. Auf diese Art können Sie Ihre Mappings in kleinere Bausteine aufteilen und diese mehrmals und auch in anderen Mappings verwenden.



Benutzerdefinierte Funktionen werden zusammen mit dem Hauptmapping in der \*.mfd-Datei gespeichert.

In einer benutzerdefinierten Funktion werden **Input-** und **Output-Komponenten** verwendet, um Informationen aus dem Hauptmapping (oder einer anderen benutzerdefinierten Funktion) an die benutzerdefinierte Funktion zu übergeben und anschließend wieder zurückzugeben.

Benutzerdefinierte Funktionen können "lokale" Quellkomponenten (also Komponenten, die sich selbst innerhalb der benutzerdefinierten Funktion befinden) wie z.B. XML-Schemas oder Datenbanken enthalten, die für die Implementierung von lookup-Funktionen nützlich sind.

Benutzerdefinierte Funktionen können aus beliebig vielen Inputs und Outputs bestehen, wobei jeder dieser Inputs und Outputs aus einfachen Werten, XML Nodes oder Datenbanken oder EDI-Dateien oder FlexText-Strukturen bestehen kann.

Benutzerdefinierte Funktionen eignen sich für folgende Zwecke:

- um mehrere Verarbeitungsfunktionen zu einer einzigen Komponente zusammenzufassen, z.B. um ein bestimmtes Feld zu formatieren oder einen Wert nachzuschlagen
- um diese Komponenten beliebig oft wiederzuverwenden
- um benutzerdefinierte Funktionen durch Laden der Mappingdatei als Bibliothek in andere Mappings [zu importieren](#).
- um ein komplexes Mapping mittels [inline Funktionen](#) in kleinere Teile aufzuteilen, die separat bearbeitet werden können
- um **rekursive Schemas** durch Erstellung von [rekursiven benutzerdefinierten Funktionen](#) zu mappen

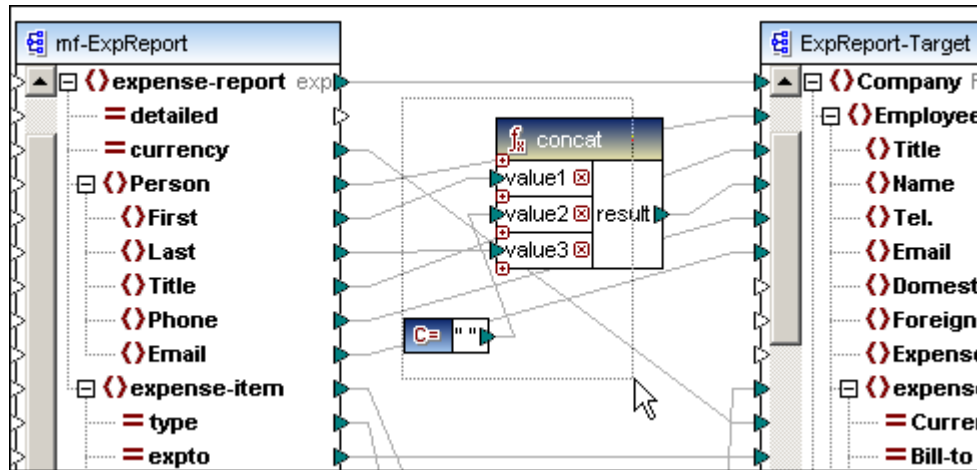
Benutzerdefinierte Funktionen können entweder von Grund auf neu erstellt werden oder aus Funktionen, die bereits auf dem Register "Mapping" verfügbar sind, zusammengesetzt werden.

In diesem Beispiel wird die Datei **Tut-ExpReport.mfd** aus dem Ordner **...\\MapForceExamples\\Tutorial** verwendet.

### So erstellen Sie eine benutzerdefinierte Funktion aus vorhandenen Komponenten:

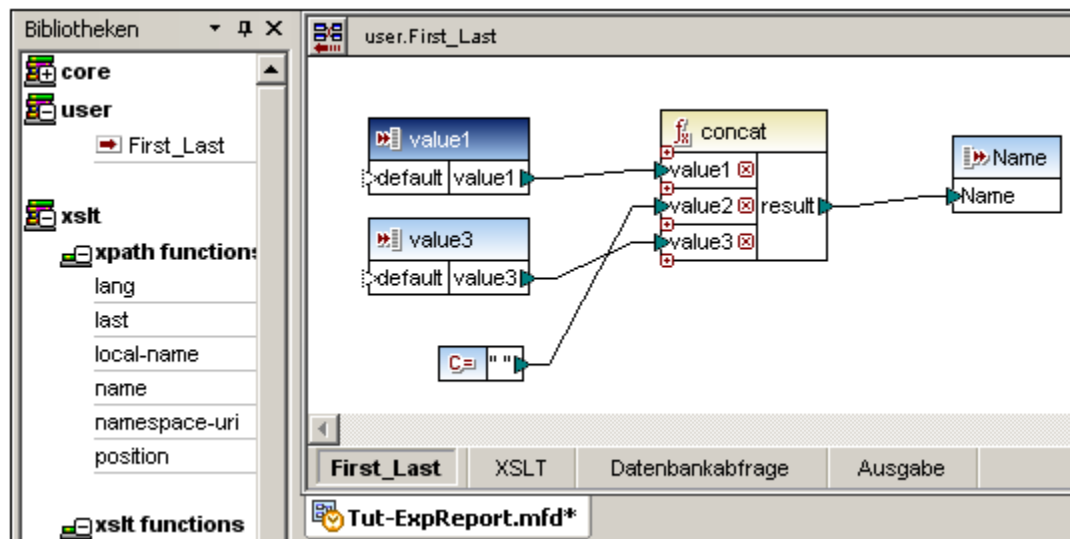
1. Ziehen Sie mit der Maus ein Rechteck über die concat-Funktion und die

constant-Funktion, um sie zu markieren (Sie können auch die Strg-Taste gedrückt halten und auf die Funktionen klicken, um diese zu markieren).



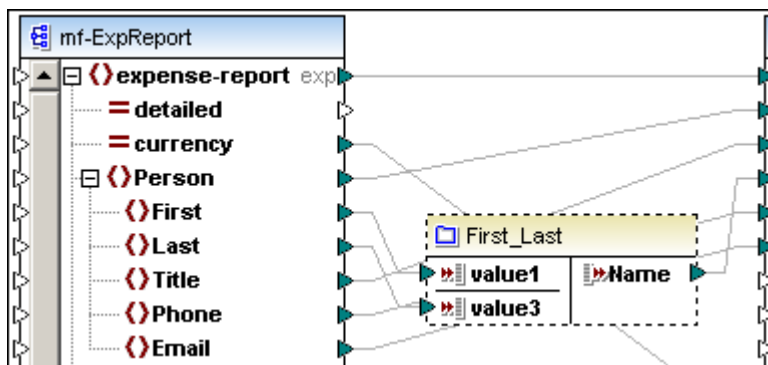
2. Wählen Sie die Menüoption **Funktion | Benutzerdefinierte Funktion von Auswahl erstellen**.
3. Geben Sie einen Namen für die neue benutzerdefinierte Funktion ein (First\_Last). Hinweis: Gültige Zeichen sind: alphanumerische, a-z, A-Z, 0-9 sowie Unterstrich "\_", Bindestrich/Schrägstrich "-" und Doppelpunkt ":".
4. Über die Felder "Syntax" und "Detail" können Sie zusätzliche Informationen zur neuen Funktion eingeben. Bestätigen Sie die Eingabe anschließend mit OK. Der von Ihnen eingegebene Text wird anschließend als Tooltip angezeigt, wenn Sie den Cursor über die Funktion platzieren. Standardmäßig wird als Bibliotheksname "user" vorgeschlagen. Sie können in diesem Feld aber natürlich Ihren eigenen Bibliotheksnamen definieren.

Die einzelnen Elemente, aus denen die Funktion besteht, werden auf einem Register mit dem Namen der Funktion angezeigt. Die neue Bibliothek "user" erscheint im Bibliotheksfenster mit dem Funktionsnamen "First\_Last" unterhalb davon.



### XSLT ausgewählt

Klicken Sie auf die Schaltfläche "Startseite" , um zum Hauptmapping-Fenster zurückzukehren. Die zwei Komponenten wurden nun zu einer einzigen Funktionskomponente mit dem Namen First\_Last zusammengefasst. Die Input- und Output-Parameter wurden automatisch verbunden.



Benutzerdefinierte inline-Funktionen werden mit einer gestrichelten Umrandung angezeigt. Im Allgemeinen verhalten sich diese Funktionen genauso, als wären ihre einzelnen Bestandteile anstatt der Funktionskomponente eingefügt worden. Nähere Informationen dazu siehe "[benutzerdefinierte inline-Funktionen](#)".

Durch Ziehen des Funktionsnamens aus dem Bibliotheksfenster in das Mapping-Fenster können Sie die Funktion an jeder Stelle des aktuellen Mappings zu verwenden. Informationen dazu, wie Sie die Funktion in einem anderen Mapping verwenden finden Sie unter [Wiederverwenden von benutzerdefinierten Funktionen](#).

### So öffnen Sie eine benutzerdefinierte Funktion:

Wählen Sie eine der folgenden Methoden:

- Doppelklicken Sie auf die Titelleiste der benutzerdefinierten Funktionskomponente oder
- Doppelklicken Sie im Bibliotheksfenster auf die jeweilige benutzerdefinierte Funktion.

Daraufhin werden die Einzelkomponenten innerhalb der Funktion auf einem Register mit diesem Namen angezeigt. Klicken Sie auf die Schaltfläche "Startseite"  um zum

Mapping zurückzukehren.

### Navigieren in benutzerdefinierten Funktionen

Wenn Sie zwischen den verschiedenen Registern (oder den Registern der benutzerdefinierten Funktionen) in MapForce navigieren, wird automatisch ein Verlauf erstellt, sodass Sie durch Klicken auf die Schaltflächen "Vorwärts" und "Zurück" zwischen den Registern vorwärts- und zurücknavigieren können. Der Verlauf gilt für die gesamte Sitzung, d.h. Sie können auch zwischen mehreren MFD-Dateien wechseln.



Über die Schaltfläche "Startseite" kehren Sie von der benutzerdefinierten Funktion aus zum Register des Hauptmappings zurück.



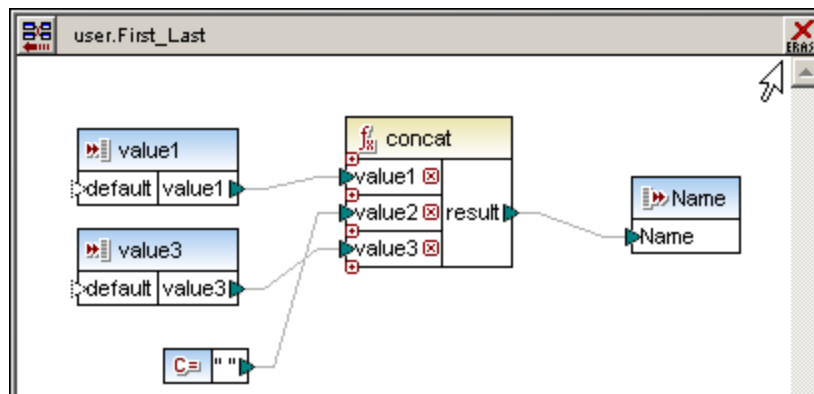
Über die Schaltfläche "Zurück" können Sie im Verlauf zurück navigieren



Über die Schaltfläche "Vorwärts" können Sie im Verlauf vorwärts navigieren

### So löschen Sie eine benutzerdefinierte Funktion aus der Bibliothek:

1. Doppelklicken Sie auf eine bestimmte benutzerdefinierte Funktion im Bibliotheksfenster. Die benutzerdefinierte Funktion erscheint nun auf dem dazugehörigen Register.
2. Klicken Sie rechts oben in der Titelleiste auf die Schaltfläche **Erase**, um die Funktion zu löschen.



### Wiederverwenden - Exportieren und Importieren von benutzerdefinierten Funktionen:

Benutzerdefinierte Funktionen, die in einem Mapping erstellt wurden, können in ein anderes Mapping importiert werden:

1. Klicken Sie am unteren Rand des Fensters "Bibliotheken" auf die Schaltfläche **Bibliotheken hinzufügen/entfernen**, klicken Sie auf die Schaltfläche "Hinzufügen" und wählen Sie eine \*.mfd-Datei aus, die die gewünschte(n) benutzerdefinierte(n) Funktion(en) enthält.

Die benutzerdefinierte Funktion wird nun im Bibliotheksfenster angezeigt (unter "user", wenn das die Standardbibliothek ist, die Sie ausgewählt haben). Sie können beim Definieren der benutzerdefinierten Funktion natürlich einen beliebigen Namen in das Feld "Bibliotheksname" eingeben.

2. Ziehen Sie die importierte Funktion in das Mapping, um sie zu verwenden.

### Bibliotheksnamen

Anmerkung: Sie können dieselbe Bibliothek für benutzerdefinierte Funktionen in mehreren \*.mfd Dateien und/oder benutzerdefinierten Bibliotheken verwenden (siehe

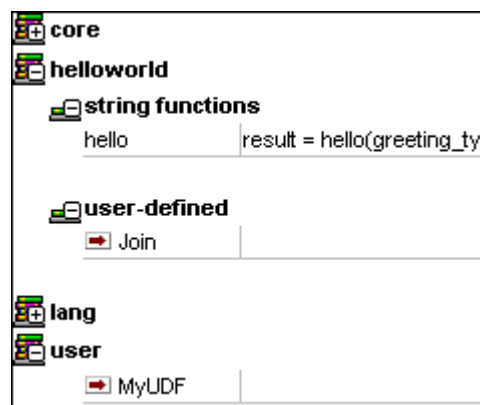


Abschnitt [Hinzufügen benutzerdefinierter Bibliotheken](#)).

Funktionen aus allen verfügbaren Quellen scheinen im Fenster "Bibliotheken" unter demselben Bibliotheksnamen auf. Doch nur die Funktionen im derzeit aktiven Dokument können durch Doppelklicken bearbeitet werden.

Im folgenden Beispiel:

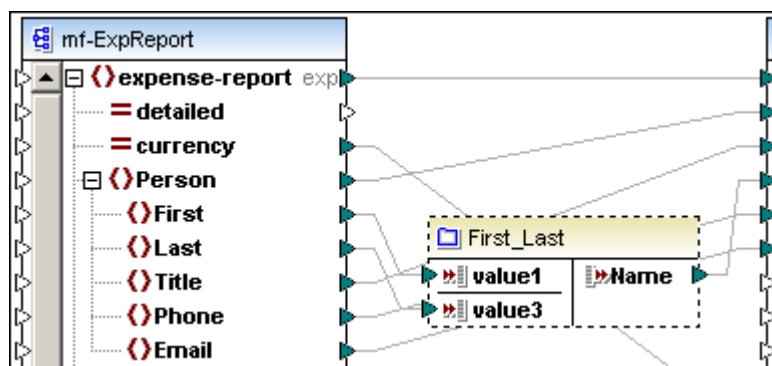
- wird die Funktion "hello" in der Bibliothek "helloworld" aus einer benutzerdefinierten \*.mff-Datei importiert.
- ist die Funktion "Join" in der Bibliothek "helloworld" eine benutzerdefinierte Funktion, die in der aktuellen \*.mfd-Datei definiert ist
- und die Funktion "MyUDF" in der Bibliothek "User" ist ebenfalls eine benutzerdefinierte Funktion, die in der aktuellen \*.mfd-Datei definiert ist.



*Java ausgewählt*

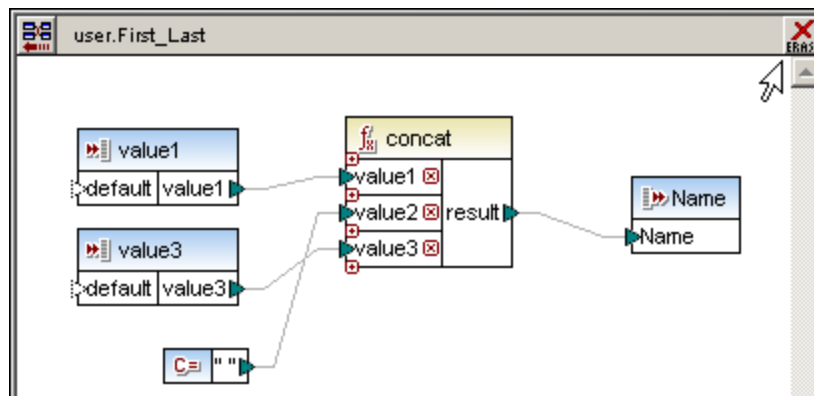
Beachten Sie, dass eventuelle Änderungen an importierten Funktionen auf importierte Mappings angewendet werden, wenn Sie die MFD-Bibliothekdatei speichern.

### Parameterreihenfolge in benutzerdefinierte Funktionen



Sie können nun auch die Parameterreihenfolge innerhalb von benutzerdefinierten Funktionen direkt beeinflussen:

- Input- und Output-Parameter werden von oben nach unten (von der linken oberen Ecke der Parameterkomponente) nach ihrer Position sortiert .
- Wenn zwei Parameter dieselbe vertikale Position haben, hat der weiter links gelegene Vorrang.
- Fall zwei Parameter ungewöhnlicherweise genau die gleiche Position haben, wird automatisch die interne Komponenten-ID verwendet.



Anmerkungen:

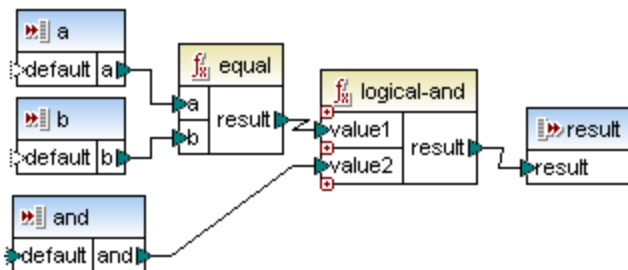
- Die Positionierung der Komponente und die Anpassung der Größe der Komponente können nun rückgängig gemacht werden.
- Neu hinzugefügte Input- oder Output-Komponenten werden unterhalb der letzten Input- oder Output-Komponente erstellt.
- Komplexe und einfache Parameter können gemischt verwendet werden. Die Reihenfolge der Parameter wird anhand der Position der Komponente ermittelt.

### 23.1.1 Funktionsparameter

Funktionsparameter werden innerhalb einer benutzerdefinierten Funktion durch **Input-** und **Output-Komponenten** dargestellt.

Input-Komponente/Parameter: **a, b and**

Output-Komponenten/Parameter: **result**

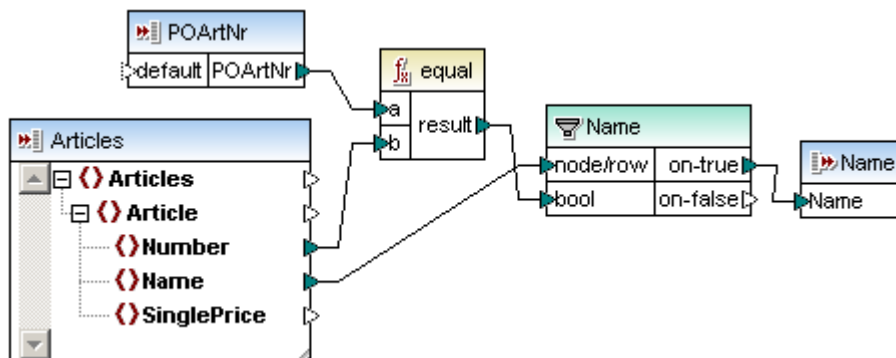


Input-Parameter dienen dazu, Informationen vom Hauptmapping an die benutzerdefinierte Funktion zu übergeben, während Output-Parameter dazu dienen, Informationen an das Hauptmapping zurückzugeben.

#### Einfache und komplexe Parameter

Die Input- und Output-Parameter von benutzerdefinierten Funktionen können eine beliebige Kombination aus folgenden Komponenten sein:

- einfachen Werten, z.B. Strings oder Ganzzahlen
- komplexen Node-Strukturen, z.B. einem XML-Element mit seinen Attributen und Child Nodes



Der Input-Parameter **POArtNr** ist ein einfacher Wert des Datentyps "string"

Der Input-Parameter **Articles** ist eine komplexe XML-Dokument Node-Struktur

Der Output-Parameter **Name** ist ein einfacher Wert vom Typ String

Anmerkung:

Die oben gezeigten benutzerdefinierten Funktionen stehen alle in der Datei **PersonListByBranchOffice.mfd** im Ordner ...\\MapForceExamples zur Verfügung.

#### Sequenzen

Sequenzen sind Quelldaten, die aus einem Bereich oder einer Sequenz von Werten bestehen. Im Dialogfeld "Komponenteneigenschaften" können einfache und komplexe benutzerdefinierte

**Parameter** (Input/Output) als Sequenzen definiert werden.

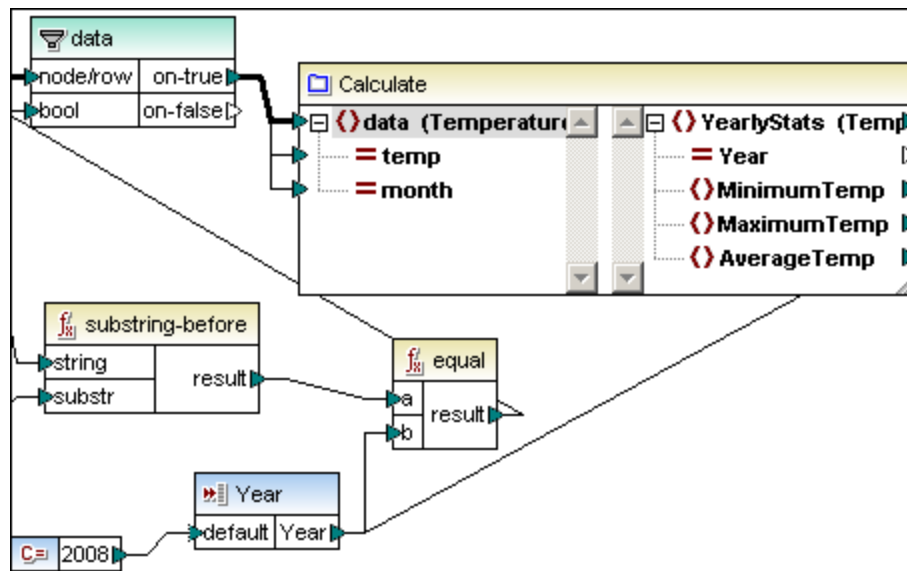
Bei Aggregatfunktionen wie z.B. min, max, avg, usw. wird diese Art von Input verwendet, um einen einzigen bestimmten Wert aus der Input-Sequenz bereitzustellen. Nähere Informationen zu Aggregatfunktionen finden Sie unter [Aggregatfunktionen: min, max, sum, count, avg](#).

Wenn das Kontrollkästchen "**Input ist eine Sequenz**" aktiv ist, so behandelt die Komponente den Input als Sequenz. Wenn das Kontrollkästchen deaktiviert ist, wird der Input als Einzelwert behandelt.

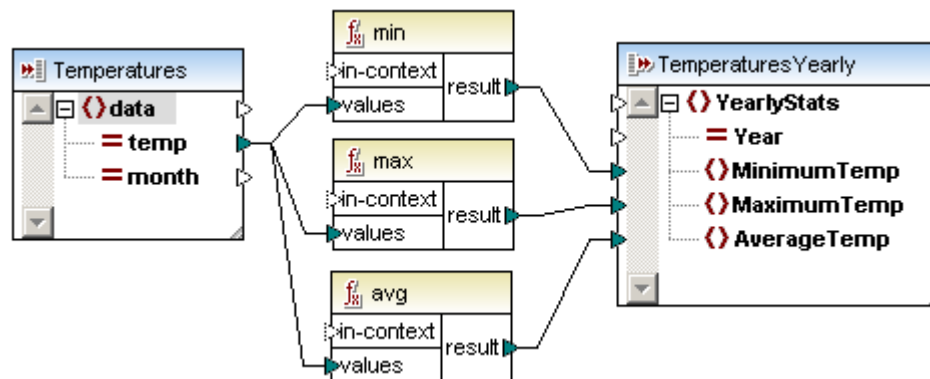
The screenshot shows the 'Input bearbeiten' (Edit Input) dialog box. The 'Name' field is set to 'Temperatures'. Under the 'Typ' (Type) section, the 'Complex Type (Baumstruktur)' radio button is selected. The 'Struktur' (Structure) field contains the path '\\MapForceExamples\\Temperatures.xsd' and the 'Root' field contains '{}Temperatures/{}data'. Both fields have an 'Auswählen' (Select) button next to them. The checkbox 'Strukturdateipfad relativ zur MFD-Datei speichern' is checked. At the bottom, the checkboxes 'Input ist erforderlich' and 'Input ist eine Sequenz' are both checked. The 'OK' and 'Abbrechen' buttons are located at the bottom right of the dialog.

Mit dieser Unterscheidung zwischen Sequenz oder nicht Sequenz wird festgelegt, **wie oft** die Funktion aufgerufen wird.

- Bei Verbindung mit einem **Sequenz**-Parameter wird die benutzerdefinierte Funktion **nur einmal** aufgerufen und die komplette Sequenz wird an die benutzerdefinierte Funktion übergeben.



In der Abbildung sehen Sie die benutzerdefinierte Funktion "Calculate" des Mappings **InputsSequence.mfd** aus dem Ordner ...\\MapForceExamples. Die Input-Komponente "Temperatures" ist, wie im obigen Dialogfeld gezeigt, als Sequenz definiert.



- Wenn Input-Daten mit einem Parameter verbunden sind, bei dem es sich **nicht** um eine Sequenz handelt, wird die Funktion für jedes einzelne Datenelement in der Sequenz **einmal** aufgerufen.

Bitte beachten Sie:

Die Sequenzeinstellung von Input/Output-Parametern wird ignoriert, wenn die Funktion vom Typ [inline](#) ist.

Wenn Sie **eine leere Sequenz** mit einem **Parameter**, verbinden, bei dem es sich um **keine Sequenz** handelt, hat dies zur Folge, dass die Funktion **gar nicht aufgerufen wird**!

Dies kann passieren, wenn die Quellstruktur **optionale** Datenelemente enthält oder wenn eine Filterbedingung keine passenden Datenelemente zurückgibt. Um dies zu vermeiden und sicherzustellen, dass die Sequenz nie leer ist, verwenden Sie entweder vor dem Input der Funktion die [substitute-missing](#) Funktion oder setzen Sie den Parameter auf **sequence** und fügen Sie innerhalb der Funktion Behandlungsroutinen für die leere Sequenz hinzu.

Wenn eine Funktion eine Sequenz mehrerer Werte an ihre Output-Komponente übergibt und die Output-Komponente nicht auf Sequenz gesetzt wurde, wird bei Aufruf der Funktion nur das

erste Ergebnis aufgerufen.

### 23.1.2 Inline-Funktionen und reguläre benutzerdefinierte Funktionen

Inline-Funktionen unterscheiden sich in der Art, wie sie bei der Generierung von Code implementiert werden, grundlegend von nicht-inline-Funktionen.

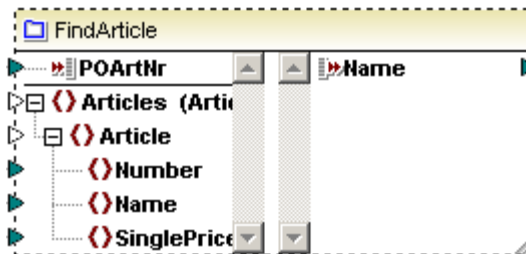
- Der Code für Funktionen vom Typ **inline** wird überall dort eingefügt, wo die benutzerdefinierten Funktionen aufgerufen/verwendet werden
- Der Code einer **regulären** Funktion wird als Funktionsaufruf implementiert.

Daher verhalten sich inline-Funktionen auf dieselbe Art, als wären sie durch ihre Implementierung ersetzt worden. Dadurch eignen sie sich ideal dazu, ein komplexes Mapping in mehrere separate Teile aufzuteilen.

Bitte beachten Sie:

Durch Verwendung von **Inline**-Funktionen kann der generierte Programmcode erheblich länger werden! Der Code der benutzerdefinierten Funktion wird an allen Stellen, an denen die Funktion aufgerufen/verwendet wird, eingefügt, wodurch der Code - im Gegensatz zur Verwendung einer normalen Funktion - erheblich länger wird.

Benutzerdefinierte **INLINE**-Funktionen werden durch eine **strichlierte** Umrandung dargestellt.



Benutzerdefinierte **Inline**-Funktionen unterstützen:

- **mehrere Output-Komponenten** innerhalb einer Funktion
- 

Benutzerdefinierte **Inline**-Funktionen unterstützen Folgendes **nicht**:

- die Festlegung eines Prioritätskontexts bei einem Parameter.
- Rekursive Aufrufe benutzerdefinierter inline-Funktionen

**REGULÄRE benutzerdefinierte Funktionen**, d.h. Funktionen, die nicht inline gesetzt wurden, werden durch eine **durchgezogene** Umrandung dargestellt



Benutzerdefinierte **reguläre** Funktionen (nicht inline gesetzte) **unterstützen**:

- lediglich eine **einzige Output-Komponente** innerhalb einer Funktion
- rekursive Aufrufe (wobei die exit-Bedingung bereitgestellt werden muss. Verwenden Sie z.B. eine if-Else-Bedingung, wenn ein Branch oder Wert die Rekursion beendet)
- das Definieren eines Prioritätskontexts bei einem Parameter

Bitte beachten Sie:

Zwar unterstützen reguläre Funktionen **nicht** mehrere Output-Komponenten, doch können sie

in dieser Funktionsart **erstellt werden**. Es erscheint allerdings eine Fehlermeldung, wenn Sie versuchen, Code zu generieren oder eine Vorschau des Ergebnisses dieses Mappings anzuzeigen.

Wenn Sie nicht mit Rekursionen in Ihren Funktionen arbeiten, können Sie die Art der Funktion in "inlined" ändern.

Benutzerdefinierte reguläre Funktionen (nicht inline gesetzte Funktionen) **unterstützen**

**Folgendes nicht:**

- die direkte Verbindung von Filtern mit einfachen **Input**-Komponenten, bei denen es sich nicht um eine Sequenz handelt
- Sequenz- oder Aggregatfunktionen zu einfachen Input-Komponenten (wie like exists, substitute-missing, sum, group-by, ...)

**Codegenerierung:**

Die Implementierung einer regulären benutzerdefinierten Funktion wird nur einmal als aufrufbare XSLT-Vorlage oder Funktion generiert. Jede benutzerdefinierte Funktion generiert Code für einen Funktionsaufruf, bei dem Inputs als Parameter übergeben werden und der Output der Rückgabewert der Funktionskomponente ist.

Die Input-Parameter werden zuerst in Laufzeit ausgewertet, anschließend wird die Funktion jedesmal aufgerufen, wenn die Input-Daten vorkommen. Nähere Informationen dazu finden Sie unter [Funktionsparameter](#).

**So ändern Sie die benutzerdefinierte Funktion "type":**

1. Doppelklicken Sie auf die benutzerdefinierte Funktion, um deren Komponenten zu sehen.
2. Wählen Sie die Menüoption **Funktion | Funktionseinstellungen** und aktivieren Sie das Kontrollkästchen "Inline Verwendung".

**Benutzerdefinierte Funktionen und "Alles kopieren" Verbindungen**

Beim Erstellen von "Alles kopieren" Verbindungen zwischen einem Schema und einem komplexen benutzerdefinierten Funktionsparameter müssen die beiden Komponenten auf demselben Schema basieren! Sie müssen jedoch nicht notwendigerweise dasselbe Root-Element haben. Ein Beispiel dazu finden Sie unter "[Definieren komplexer Output-Komponenten](#)".



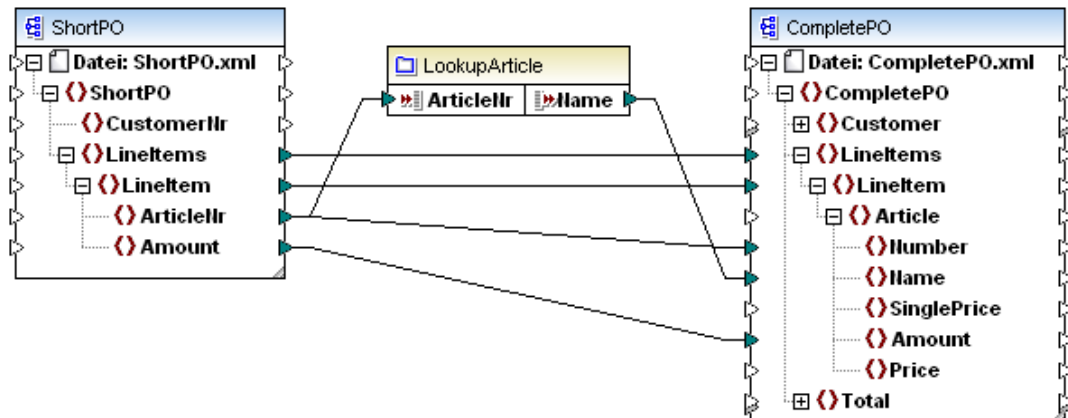
### 23.1.3 Erstellen einer einfachen Lookup-Funktion

In diesem Beispiel werden die Komponenten aus der Datei **lookup-standard.mfd** verwendet, die Sie im Ordner **...MapForceExamples** finden.

Ziel:

Die Erstellung einer generischen Abfragefunktion, die:

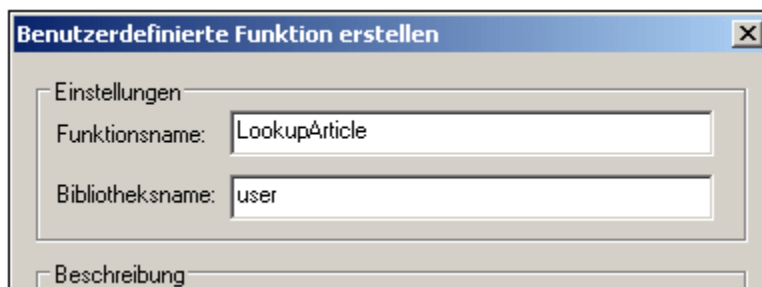
- die Artikel-/Nummerdaten aus der XML-Artikeldatei liefert, die mit den Artikelnummern einer anderen XML-Datei - in diesem Fall ShortPO - verglichen werden.



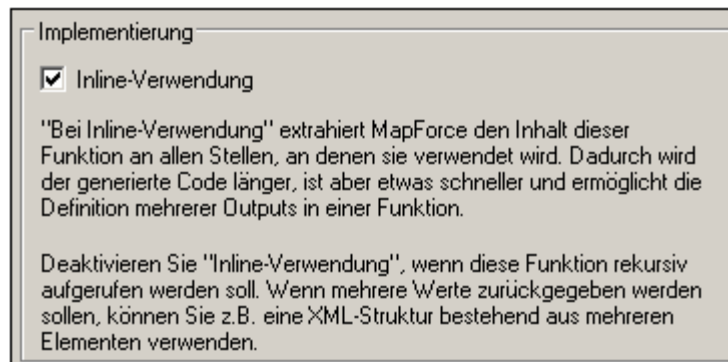
- Fügen Sie die Datei ShortPO.xml ein und weisen Sie dieser ShortPO.xml als XML-Quelldatei zu.
- Fügen Sie die Schema-Datei CompletePO.xsd ein und wählen Sie als Root-Element CompletePO.
- Fügen Sie auf die unten beschriebene Weise eine neue benutzerdefinierte Funktion ein.

**So erstellen Sie eine benutzerdefinierte Funktion von Grund auf neu:**

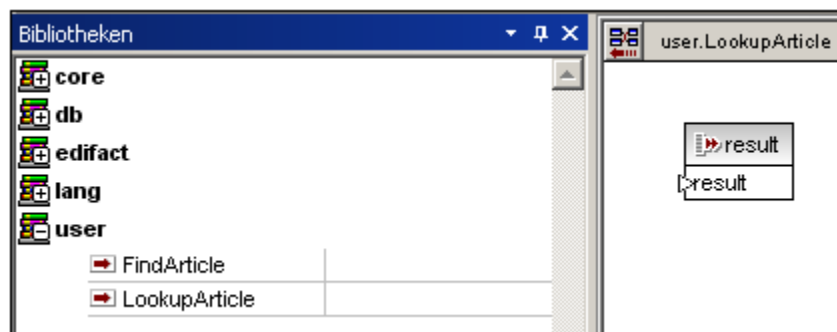
1. Wählen Sie die Menüoption **Funktion | Benutzerdefinierte Funktion erstellen**.
2. Geben Sie einen Namen für die Funktion ein, z.B. LookupArticle.



3. Deaktivieren Sie das Optionsfeld "Inline Verwendung" und bestätigen Sie mit OK.



Daraufhin wird eine Registerkarte angezeigt, die nur ein Element, nämlich eine Ausgabefunktion mit dem Namen "result", enthält.



*Java ausgewählt*

Dies ist der Arbeitsbereich, in dem die benutzerdefinierte Funktion definiert wird.

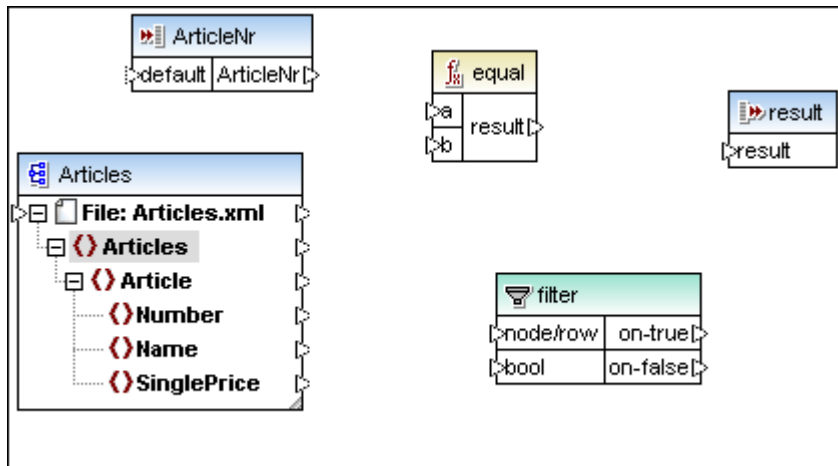
Im Bibliotheksfenster wurde eine neue Bibliothek mit dem Namen "user" angelegt, die die Funktion mit den Titel "LookupArticle" enthält.

4. Klicken Sie auf das Symbol **XML-Schema/Datei einfügen** , um das Schema **Articles** einzufügen und wählen Sie die XML-Datei desselben Namens als Datenquelle aus.
5. Klicken Sie auf das Symbol **Input-Komponente einfügen** , um eine Input-Komponente einzufügen.
6. Geben Sie den Namen des Eingabeparameters ein, in diesem Fall ArticleNr und klicken Sie auf OK.

Diese Komponente dient als Dateninput für die benutzerdefinierte Funktion und stellt das Input-Symbol der benutzerdefinierten Funktion bereit.

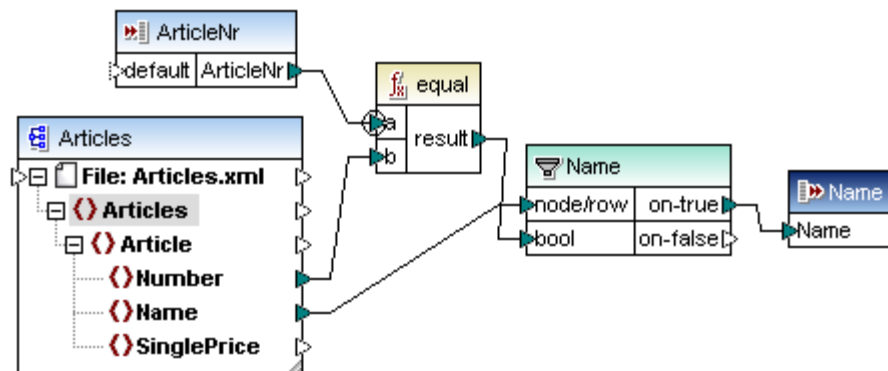
Sie können den Datentyp des Eingabeparameters ändern und definieren, ob die Funktion ein Input-Symbol in diesem Dialogfeld haben soll (Verbindung erforderlich).

7. Fügen Sie eine **"equal"**-Komponente ein, indem Sie sie aus der Gruppe der logical functions der core-Bibliothek auf das Register ziehen.
8. Fügen Sie eine **"filter"**-Komponente ein. Klicken Sie dazu auf das Filter-Symbol  in der Symbolleiste.



Nehmen Sie beim Erstellen der Mappings in der benutzerdefinierten Funktion das unten stehende Diagramm zu Hilfe und beachten Sie bitte Folgendes:

9. Klicken Sie mit der rechten Maustaste auf den Parameter **a** und wählen Sie im Popup-Menü den Eintrag **Priorität - Kontext** wählen.
10. **Doppelklicken** Sie auf die Ausgabefunktion und geben Sie den Namen des Ausgabeparameters ein, in diesem Fall **"Name"**.



Damit ist die Definition der benutzerdefinierten Funktion abgeschlossen.

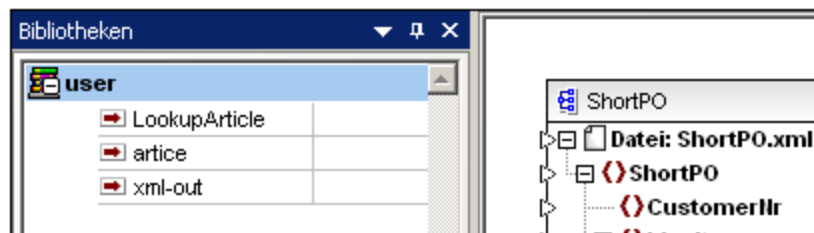
Bitte beachten Sie:

Wenn Sie auf die Eingabe- und die Ausgabefunktion doppelklicken, wird ein Dialogfeld geöffnet, in dem Sie den Namen und Datentyp des Input-Parameters ändern und definieren können, ob die Funktion ein Input-Symbol (Input erforderlich) haben soll und ob sie außerdem als Sequenz definiert werden soll.

Diese benutzerdefinierte Funktion:

- hat eine **Eingabefunktion**, ArticleNr, die Daten von der XML-Datei ShortPO erhält.
- **vergleicht** die Artikelnummer (ArticleNr) aus der Datei ShortPO mit dem Artikel/der Nummer aus der XML-Instanzdatei **Articles**, die zu diesem Zweck in die benutzerdefinierte Funktion eingefügt wurde.
- verwendet eine **filter**-Komponente, um die Datensätze Article/Name an die Ausgabekomponente zu übergeben, wenn das Ergebnis des Vergleichs "true" ist.
- hat eine Ausgabefunktion, Name, die die Datensätze mit den Artikelnamen an die XML-Datei CompletePO übergibt.

8. Klicken Sie auf das Symbol "Startseite" , um zum Haupt-Mapping zurückzukehren. Die benutzerdefinierte Funktion "LookupArticle" steht nun in der user-Bibliothek zur Verfügung.

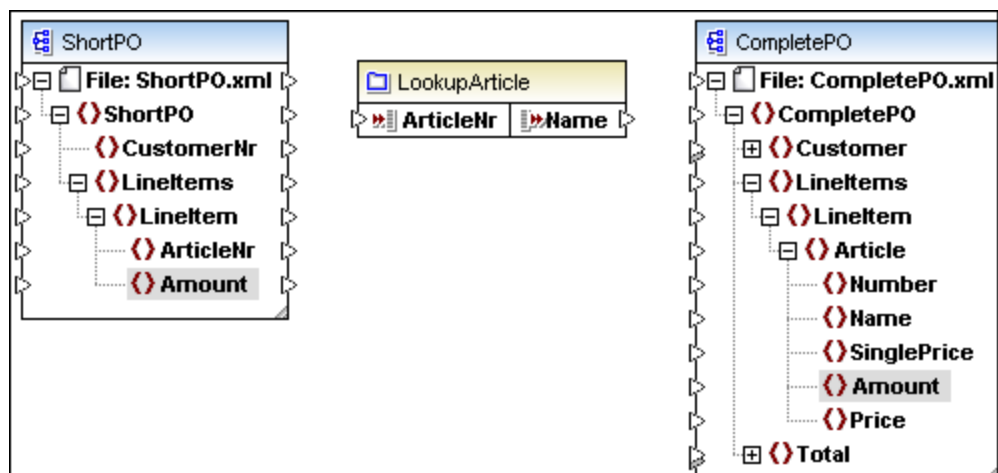


**Java ausgewählt**

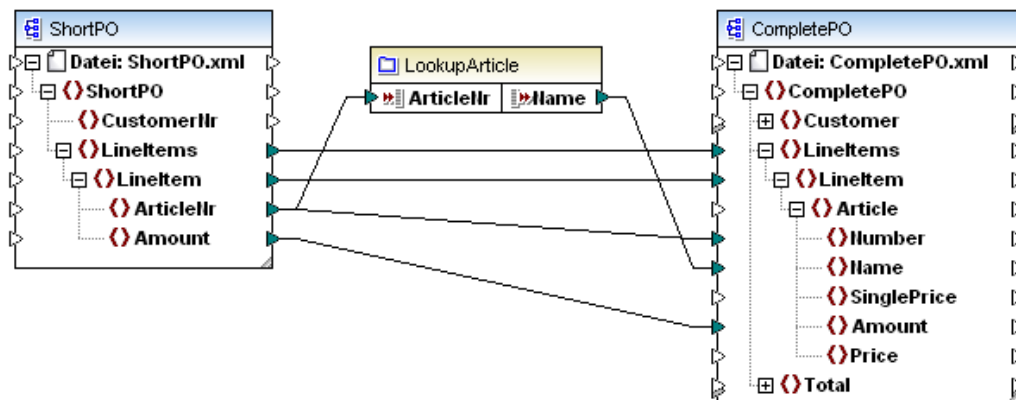
9. Ziehen Sie die Funktion LookupArticle in das Mapping-Fenster.

Die benutzerdefinierte Funktion wird angezeigt:

- Ihr Name "LookupArticle" erscheint in der Funktionstitelleiste,
- das Input- und das Output-Symbol sind mit einem Namen versehen.



10. Erstellen Sie die in der Abbildung unten gezeigten Verbindungen und klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.



### 23.1.4 Komplexe benutzerdefinierte Funktion - XML-Node als Input

Für dieses Beispiel wird die Datei **lookup-udf-in.mfd** aus dem Ordner **...MapForceExamples** verwendet. In diesem Abschnitt wird beschrieben, wie man benutzerdefinierte Inline-Funktionen definiert, die komplexe Input- und Output-Komponenten enthalten.

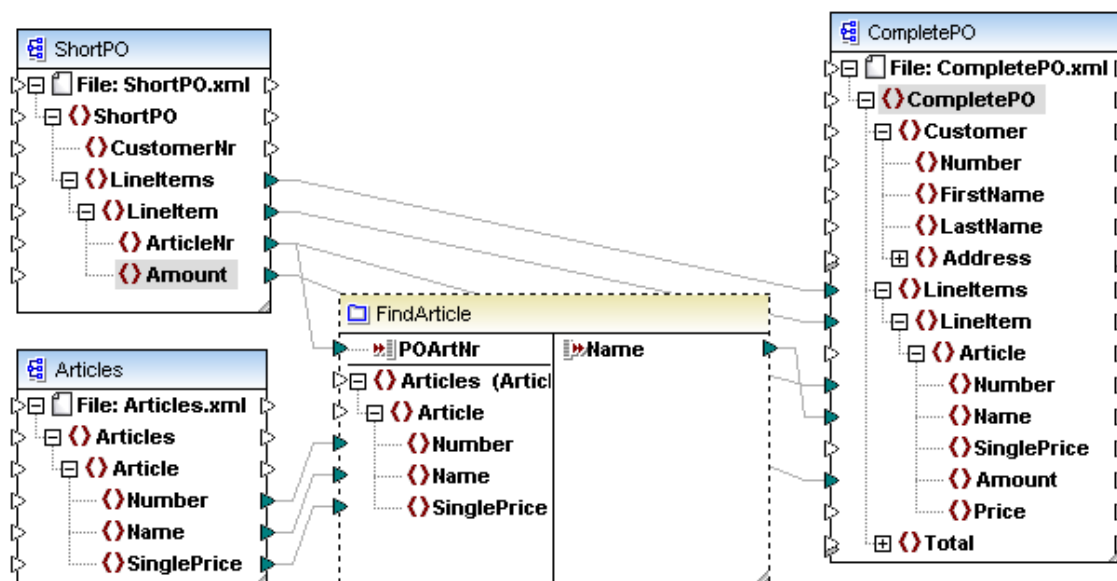
Beachten Sie, dass die benutzerdefinierte Funktion FindArticle aus zwei Hälften besteht:

einer linken Hälfte, die die Input-Parameter enthält:

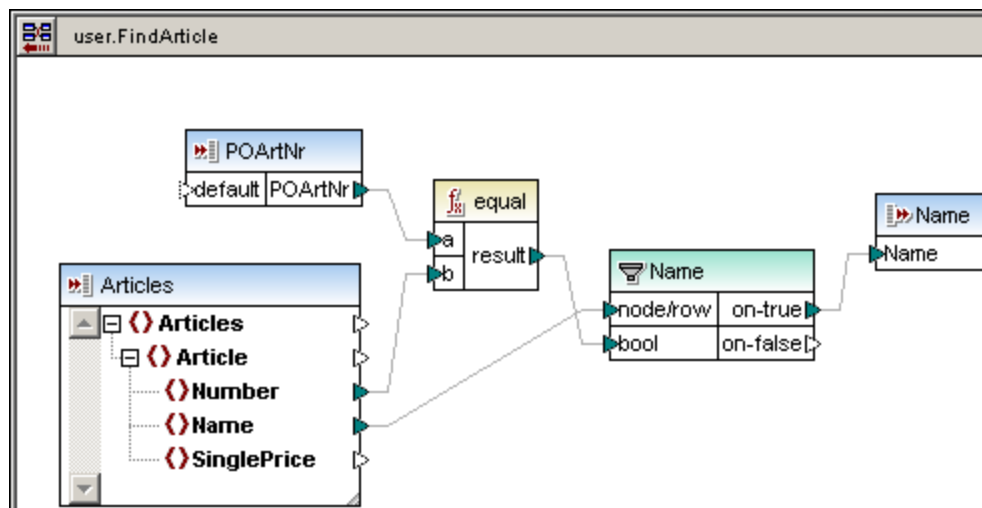
- einen einfachen Input-Parameter **POArtNr**
- eine komplexe Input-Komponente **Articles** mit Mappings direkt zu ihren XML Child Nodes

eine rechte Hälfte mit folgendem Inhalt:

- einen einfachen Output-Parameter namens **"Name"**.



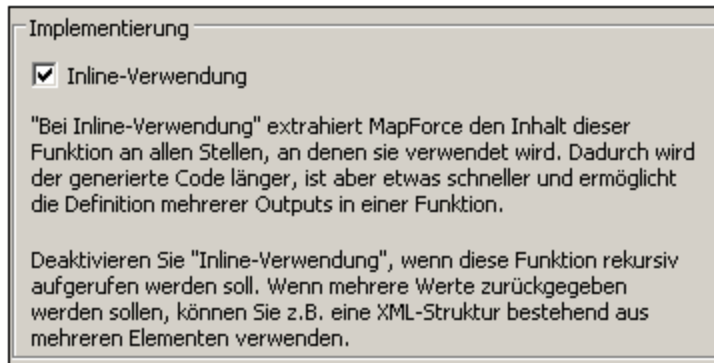
Im Screenshot unten sehen Sie die Komponenten, aus denen sich die benutzerdefinierte Funktion zusammensetzt: den beiden Input-Komponenten auf der linken Seite und der Output-Komponente auf der rechten Seite.



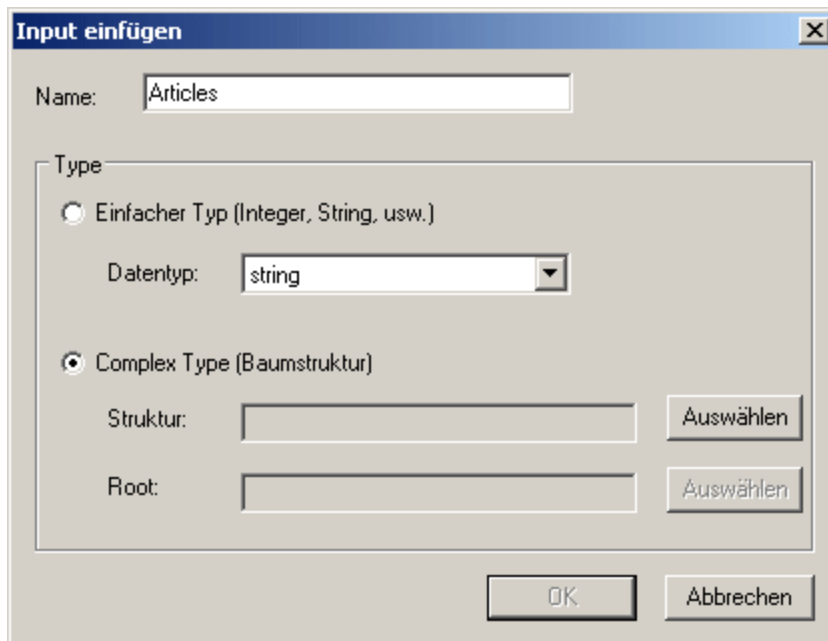
## Definieren komplexer Input-Komponenten

### Definieren komplexer Input-Komponenten:

1. Erstellen Sie wie gewohnt, also durch Auswahl von **Funktion | Benutzerdefinierte Funktion erstellen** eine benutzerdefinierte Funktion und klicken Sie auf OK. Beachten Sie, dass automatisch das Kontrollkästchen **Inline Verwendung** ausgewählt wird.



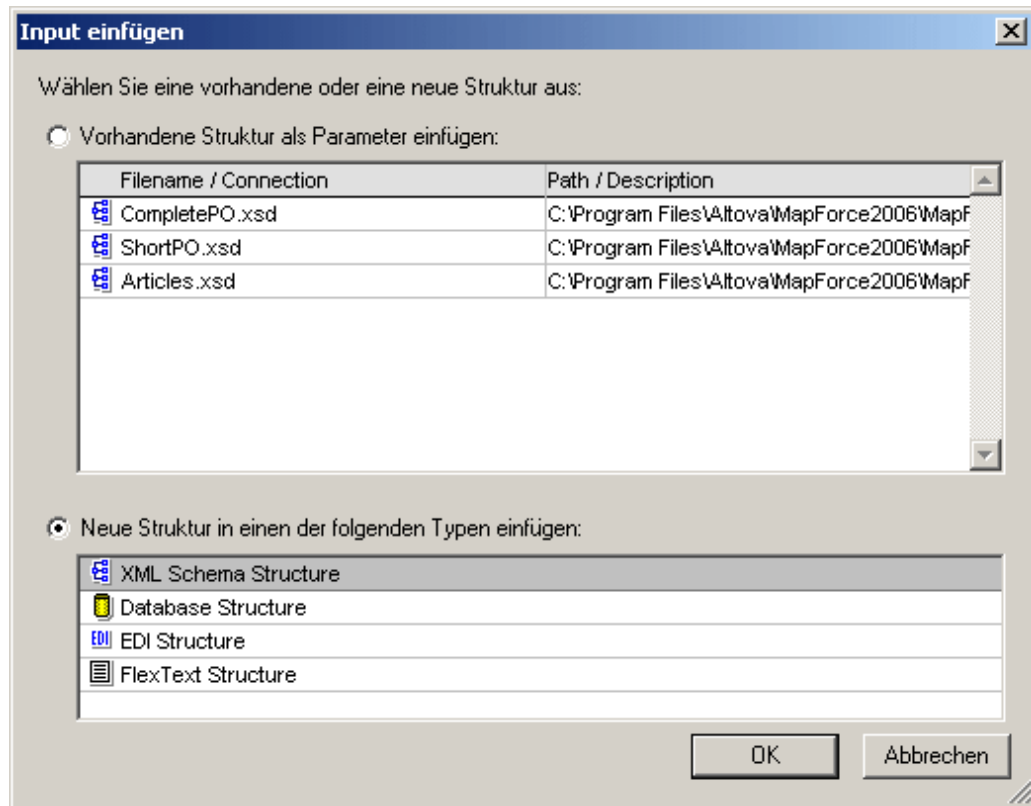
2. Klicken Sie in der Symbolleiste auf das Symbol **Input-Komponente einfügen** .
3. Geben Sie den Namen der Input-Komponente in das Feld "Name" ein.



4. Klicken Sie auf das Optionsfeld **Complex Type (Baumstruktur)** und anschließend auf die Schaltfläche "Auswählen" neben dem Feld "Struktur". Daraufhin wird ein weiteres Dialogfeld geöffnet.

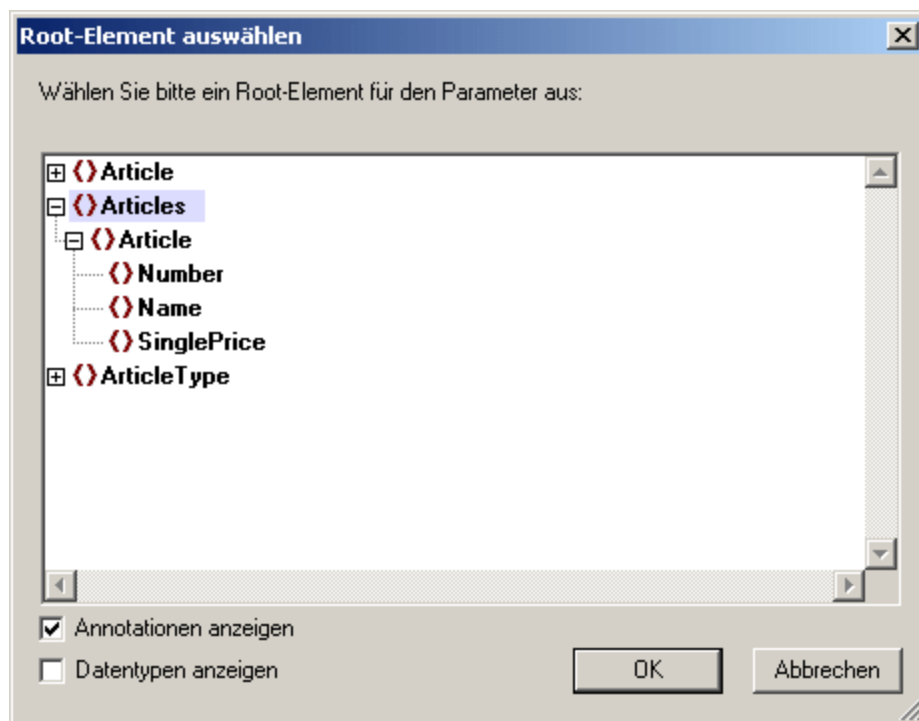
Im oberen Listenfeld sehen Sie die **vorhandenen** Komponenten im Mapping (drei Schemas, wenn Sie das Beispiel-Mapping geöffnet haben). Beachten Sie, dass diese Liste alle Komponenten enthält, die in das aktive Mapping eingefügt wurden: z.B. XML-Schemas, Datenbanken, EDI-Dateien oder FlexText-Strukturen.

Im unteren Listenfeld können Sie eine neue komplexe Datenstruktur auswählen, d.h. XML-Schema, Datenbankdatei, EDI-Datei oder FlexText-Struktur.

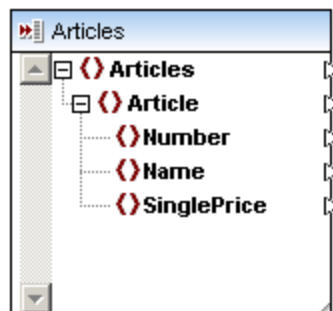


5. Klicken Sie auf das Optionsfeld "Neue Struktur einfügen...", wählen Sie den Eintrag "XML-Schema-Struktur" aus und klicken Sie zum Fortfahren auf OK.
6. Wählen Sie im Dialogfeld "Öffnen" die Datei **Articles.xsd** aus.
7. Klicken Sie auf das Element, das als Root-Element in der Komponente verwendet werden soll, z.B. Articles und bestätigen Sie die Auswahl durch Klicken auf OK. Klicken Sie anschließend nochmals auf OK, um beide Dialogfelder zu schließen.

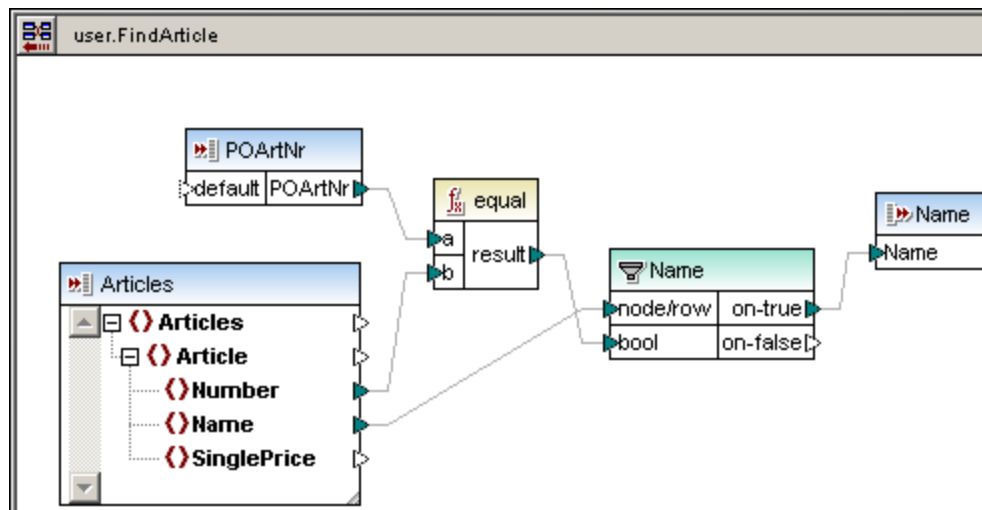




Die Komponente "Articles" wird in die benutzerdefinierte Funktion eingefügt. Beachten Sie bitte das Input-Symbol  links vom Komponentennamen. Dieses Symbol zeigt an, dass die Komponente als komplexe Input-Komponente verwendet wird.



8. Fügen Sie die restlichen Komponenten, nämlich eine zweite "einfache" Input-Komponente (namens POArtNr), Filter-, equal- und eine Output-Komponente (namens "Name") ein, wie im Screenshot unten gezeigt, und verbinden Sie diese wie gezeigt.



Bitte beachten Sie:

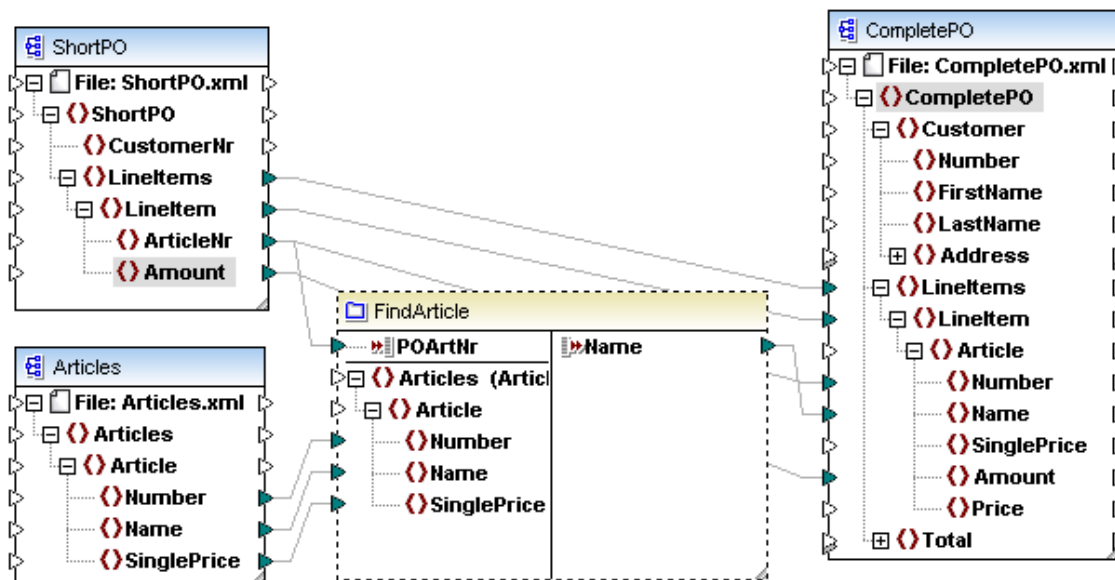
- Die Input-Komponente "Articles" erhält ihre Daten von **außerhalb** der benutzerdefinierten Funktion. Hier stehen Input-Symbole zur Verfügung, die ein Mapping auf diese Komponente gestatten.
- Sie können einer komplexen Input-Komponente keine **XML-Instanzdatei** zuweisen, um Daten aus der benutzerdefinierten Funktion bereitzustellen.
- Die andere Input-Komponente (POArtNr) liefert die Daten für die ShortPO-Artikelnnummer, mit der die Elemente **Article | Number** verglichen werden.
- Die Filterkomponente filtert jene Datensätze, in denen beide Nummern identisch sind, und übergibt diese an die Output-Komponente.

10. Klicken Sie auf das Startseitensymbol  um zum Mapping zurückzukehren.
11. Ziehen Sie die neu erstellte benutzerdefinierte Komponente aus dem Bibliotheksfenster in das Mapping.



**Java ausgewählt**

12. Erstellen Sie die Verbindungen wie im Screenshot unten gezeigt.



Die linke Hälfte enthält die Input-Parameter, auf die Datenelemente aus zwei Schema-/XML-Dateien gemappt werden:

- ShortPO liefert die Daten für die Input-Komponente **POArtNr**
- Articles stellt die Daten für die komplexe Input-Komponente bereit. Die Instanzdatei Articles.xml wurde der Schema-Datei Articles beim Einfügen dieser Datei zugewiesen.
- Die komplexe Input-Komponente **Articles** mit ihren XML Child Nodes, auf die Daten aus der Komponente "Articles" gemappt wurden.

Die rechte Hälfte enthält:

- einen einfachen Output-Parameter mit der Bezeichnung **"Name"**, der die gefilterten "Line Items", die dieselbe Artikelnummer haben, an das Datenelement "Name" in der Komponenten "CompletePO" übergibt.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 <LineItems>
4 <LineItem>
5 <Article>
6 <Number>3</Number>
7 <Name>Pants</Name>
8 <Amount>5</Amount>
9 </Article>
10 </LineItem>
11 <LineItem>
12 <Article>
13 <Number>1</Number>
14 <Name>T-Shirt</Name>
15 <Amount>17</Amount>
16 </Article>
17 </LineItem>
18 </LineItems>
19 </CompletePO>

```

Bitte beachten Sie:

Beim Erstellen von "Alles kopieren" Verbindungen zwischen einem Schema und einem benutzerdefinierten Funktionsparameter müssen die beiden Komponenten auf demselben Schema basieren! Sie müssen jedoch nicht unbedingt dasselbe Root-Element haben.

### 23.1.5 Komplexe benutzerdefinierte Funktion - XML Node als Output

Für dieses Beispiel wird die Datei **lookup-udf-out.mfd** aus dem Ordner **...MapForceExamples** verwendet. In diesem Abschnitt wird beschrieben, wie man benutzerdefinierte Inline-Funktionen definiert, die komplexe Input- und Output-Komponenten möglich machen.

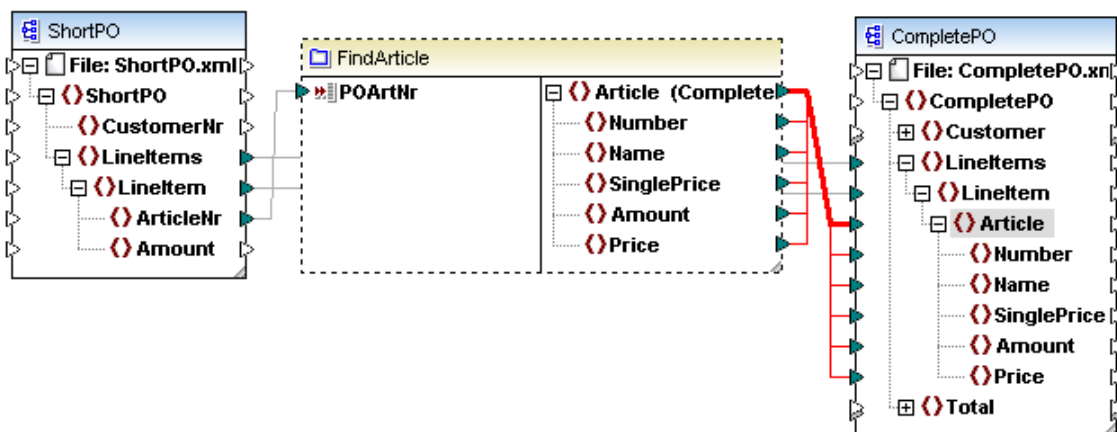
Beachten Sie, dass die benutzerdefinierte Funktion FindArticle aus zwei Hälften besteht:

einer linken Hälfte, die den Input-Parameter enthält:

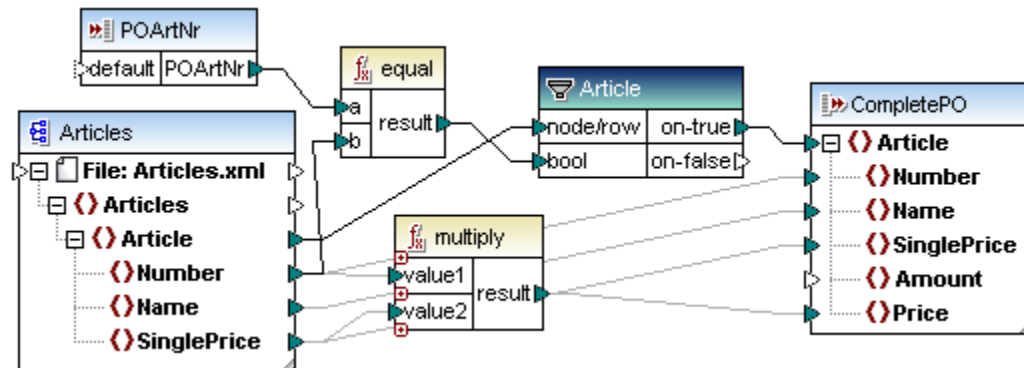
- einen einfachen Input-Parameter **POArtNr**

eine rechte Hälfte mit folgendem Inhalt:

- eine komplexe Output-Komponente namens **Article (CompletePO)**, deren XML Child Nodes auf CompletePO gemappt sind.



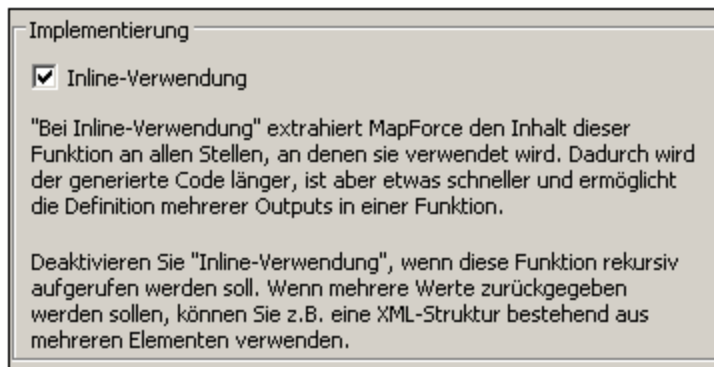
Im Screenshot unten sehen Sie die Komponenten, aus denen sich die benutzerdefinierte Funktion zusammensetzt: den beiden Input-Komponenten auf der linken Seite und der Output-Komponente auf der rechten Seite.



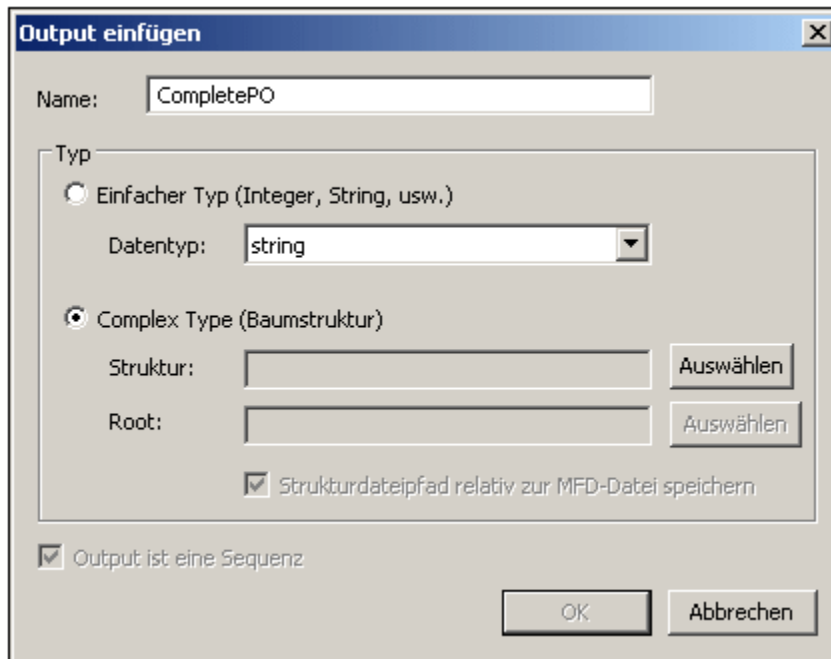
#### Definieren komplexer Output-Komponenten

##### Definieren komplexer Output-Komponenten:

1. Erstellen Sie wie gewohnt, also durch Auswahl von **Funktion | Benutzerdefinierte Funktion erstellen** eine benutzerdefinierte Funktion, geben Sie ihr den Namen **FindArticle** und bestätigen Sie den Vorgang durch Klicken auf OK. Beachten Sie, dass automatisch die Option **Inline...** ausgewählt wird.



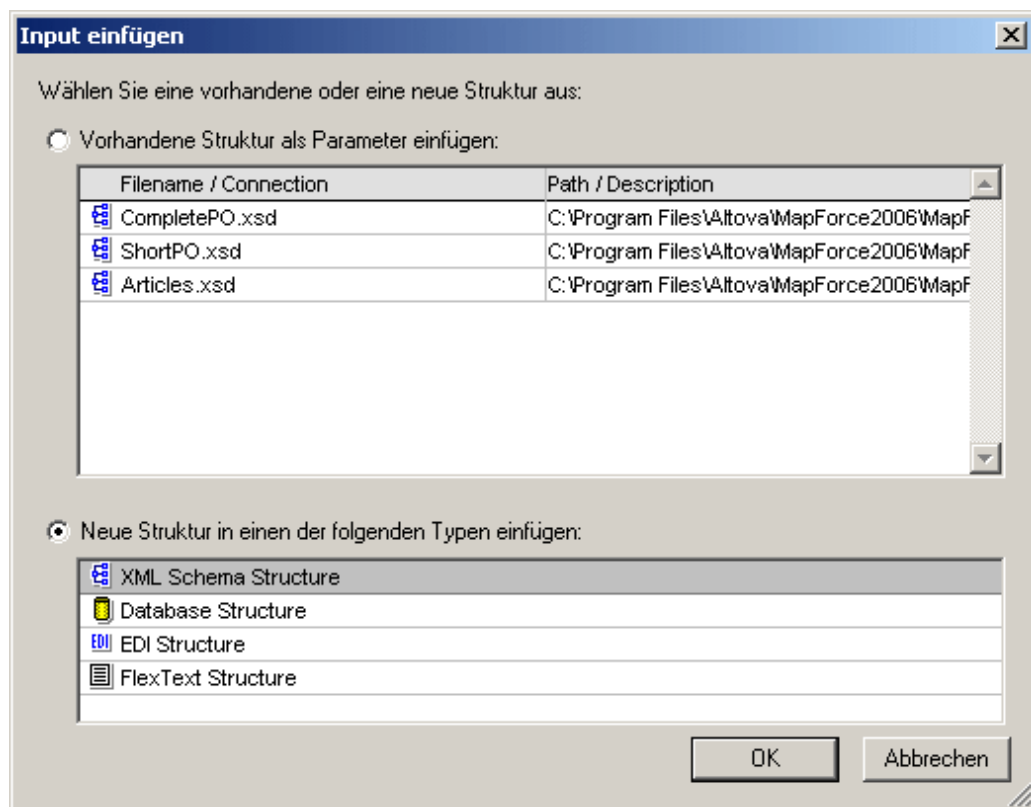
2. Klicken Sie auf das Symbol "Output-Komponente einfügen"  und geben Sie einen Namen ein, z.B. CompletePO.



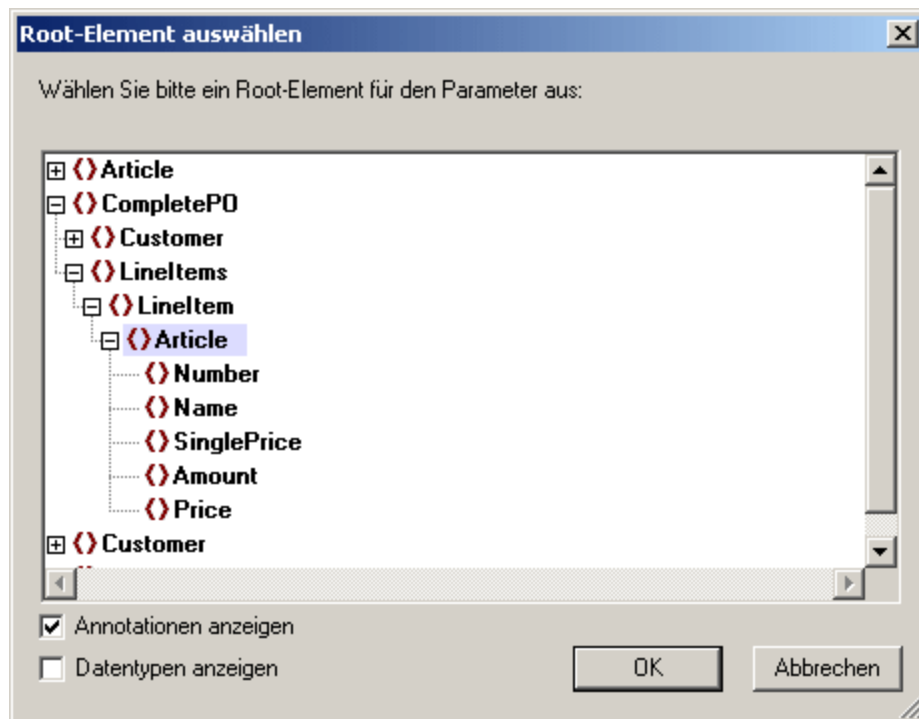
3. Klicken Sie auf das Optionsfeld **Complex type...** und anschließend auf die Schaltfläche "Auswählen".  
Daraufhin wird ein weiteres Dialogfeld geöffnet.

Im oberen Listenfeld sehen Sie die **vorhandenen** Komponenten im Mapping, (drei Schemas, wenn Sie die Beispieldatei geöffnet haben). Beachten Sie, dass diese Liste alle Komponenten enthält, die in das aktive Mapping eingefügt wurden: z.B. XML-Schemas, Datenbanken, EDI-Dateien oder FlexText-Strukturen.

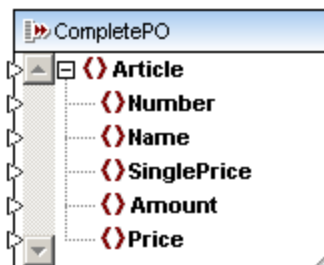
Im unteren Listenfeld können Sie eine neue komplexe Datenstruktur auswählen, d.h. XML-Schema, Datenbankdatei, EDI-Datei oder FlexText-Struktur.



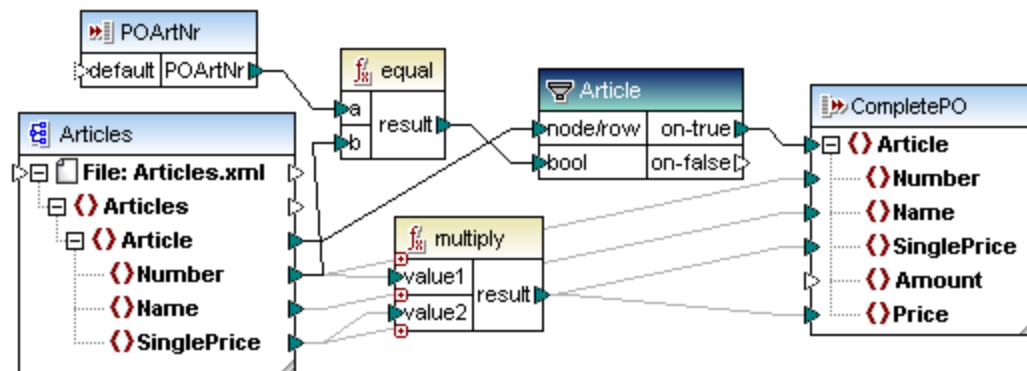
4. Klicken Sie auf das Optionsfeld "Neue Struktur einfügen...", wählen Sie den Eintrag "XML-Schema-Struktur" aus und klicken Sie zum Fortfahren auf OK.
5. Wählen Sie im Dialogfeld "Öffnen" die Datei CompletePO.xsd aus.
6. Klicken Sie auf das Element, das als Root-Element in der Komponente verwendet werden soll, z.B. **Article** und bestätigen Sie die Auswahl durch Klicken auf OK. Klicken Sie anschließend nochmals auf OK, um die Dialogfelder zu schließen.



Die Komponente "CompletePO" wird in die benutzerdefinierte Funktion eingefügt. Beachten Sie bitte das Output-Symbol  links vom Komponentennamen. Dieses Symbol zeigt an, dass die Komponente als komplexe Output-Komponente verwendet wird.



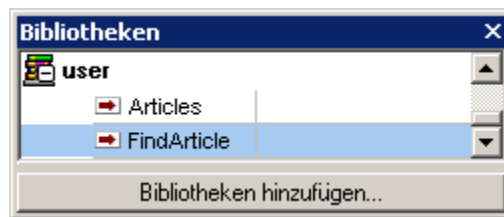
7. Fügen Sie das Schema-/die XML-Datei Articles in die benutzerdefinierte Funktion ein und weisen Sie die Datei Articles.xml als XML-Input-Instanz zu.
8. Fügen Sie die restlichen Komponenten wie im Screenshot unten gezeigt hinzu, nämlich: zwei "einfache" Input-Komponenten (POArtNr) sowie Filter-, equal- und multiply-Komponenten und verbinden Sie diese wie gezeigt.



Bitte beachten Sie:

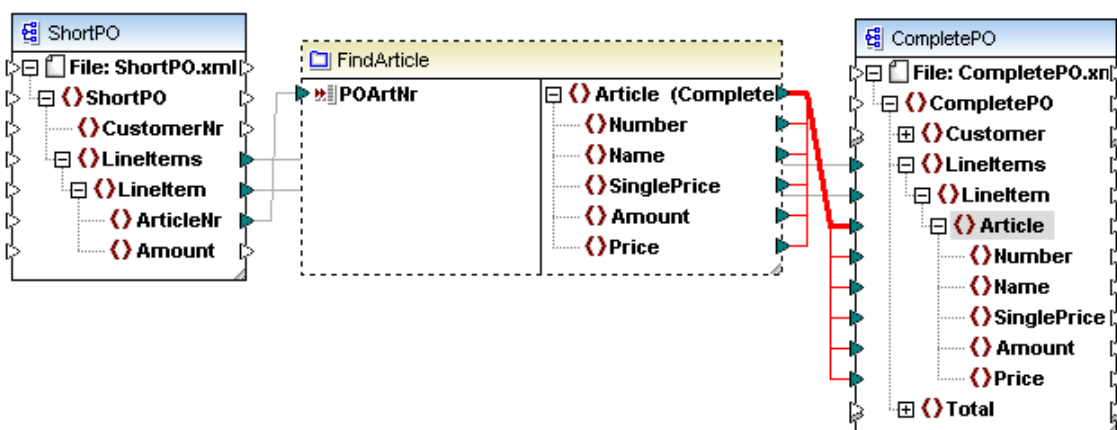
- Die Komponente "Articles" erhält ihre Daten aus der Beispieldatei Articles.xml in der benutzerdefinierten Funktion.
- Die Input-Komponenten stellen die Artikelnummer (POArtNr) und die Amount-Daten bereit, mit denen Articles | Number & Price verglichen werden.
- Die Filterkomponente filtert jene Datensätze, in denen die beiden Nummern identisch sind und übergibt sie an die Output-Komponente CompletePO.

9. Klicken Sie auf das Startseite-Symbol  um zum Mapping zurückzukehren.
10. Ziehen Sie die neu erstellte benutzerdefinierte Funktion aus dem Bibliotheksfenster in das Mapping.



*Java ausgewählt*

11. Erstellen Sie die Verbindungen, wie im Screenshot unten gezeigt. Nachdem Sie den Article (CompletePO) Konnektor zur Zielkomponente erstellt haben, klicken Sie mit der rechten Maustaste darauf und wählen Sie im Kontextmenü die Option "Alles kopieren" aus. Die restlichen Konnektoren werden automatisch generiert und sind im Screenshot unten markiert.





Bitte beachten Sie:

Beim Erstellen von "Alles kopieren" Verbindungen zwischen einem Schema und einer benutzerdefinierten Funktion vom Typ "Inline" müssen die beiden Komponenten auf demselben Schema basieren! Sie müssen jedoch nicht unbedingt dasselbe Root-Element haben.

Die linke Hälfte enthält die Input-Parameter, auf die ein einziges Datenelement gemappt wurde:

- ShortPO liefert die Artikelnummer für die Input-Komponente **POArtNr**

Die rechte Hälfte enthält:

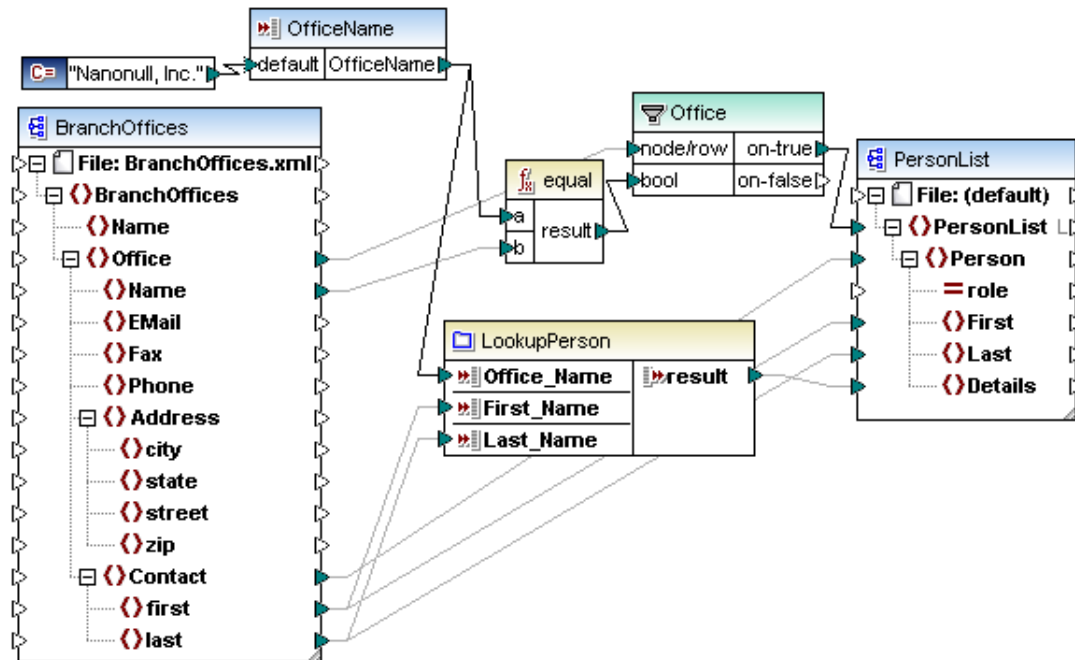
- eine komplexe Output-Komponente mit der Bezeichnung "Article (CompletePO)" mit ihren XML Child Nodes, die die gefilterten Datenelemente derselben Artikelnummer auf CompletePO mappt.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3 <LinItems>
4 <LinItem>
5 <Article>
6 <Number>3</Number>
7 <Name>Pants</Name>
8 <SinglePrice>34</SinglePrice>
9 <Price>102</Price>
10 </Article>
11 </LinItem>
12 <LinItem>
13 <Article>
14 <Number>1</Number>
15 <Name>T-Shirt</Name>
16 <SinglePrice>25</SinglePrice>
17 <Price>25</Price>
18 </Article>
```

### 23.1.6 Benutzerdefinierte Funktion - Beispiel

Anhand der Beispieldatei **PersonListByBranchOffice.mfd** aus dem Ordner **...MapForceExamples** werden die folgenden Features näher erläutert:

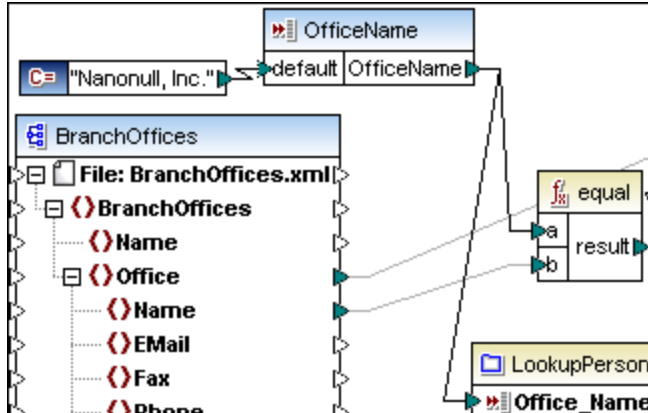
- Geschachtelte benutzerdefinierte Funktionen, z.B. **LookupPerson**
- Abfragefunktionen, die eine Ausgabe in Form eines Strings erzeugen, z.B. **LookupPerson**
- **Optionale** Eingabeparameter, die auch einen **default**-Wert liefern können, z.B. die **EqualAnd**-Komponente (enthalten in der LookupPerson-Komponente)
- **Konfigurierbare** Eingabeparameter, die auch als Befehlszeilenparameter dupliziert werden können, wenn der generierte Mapping-Code ausgeführt wird!



### Konfigurierbare Eingabeparameter

Die Eingabekomponente (OfficeName) erhält Daten, wenn ein Mapping ausgeführt wird. Dies kann auf zwei Arten erfolgen:

- als ein Befehlszeilenparameter, wenn der generierte Code, z.B. Mapping.exe/OfficeName "Nanonull Partners, Inc." ausgeführt wird.
- als Vorschauwert, wenn Sie den Built-In-Ausführungsprozessor verwenden, um eine Vorschau der Daten im Fenster "Ausgabe" anzuzeigen.



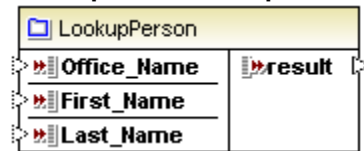
### So definieren Sie den Eingabewert:

1. Doppelklicken Sie auf die Input-Komponente und geben Sie in der Ausgabe in das Textfeld "Wert" einen anderen Wert ein, z.B. "Nanonull Partners, Inc." und bestätigen Sie mit OK.
2. Klicken Sie auf die Schaltfläche "Ausgabe", um die Auswirkungen zu sehen. Es wird nun eine andere Gruppe von Personen angezeigt.

Beachten Sie bitte, dass die in dieses Dialogfeld eingegebenen Daten nur im Vorschaufenster bzw. bei der Codegenerierung verwendet werden. Wurde kein Wert eingegeben oder ist das Kontrollkästchen deaktiviert, werden die auf das Input-Symbol "default" gemappten Daten verwendet.

Nähere Informationen dazu finden Sie unter "[Befehlszeile - Definieren von Input/Output-Dateien](#)".

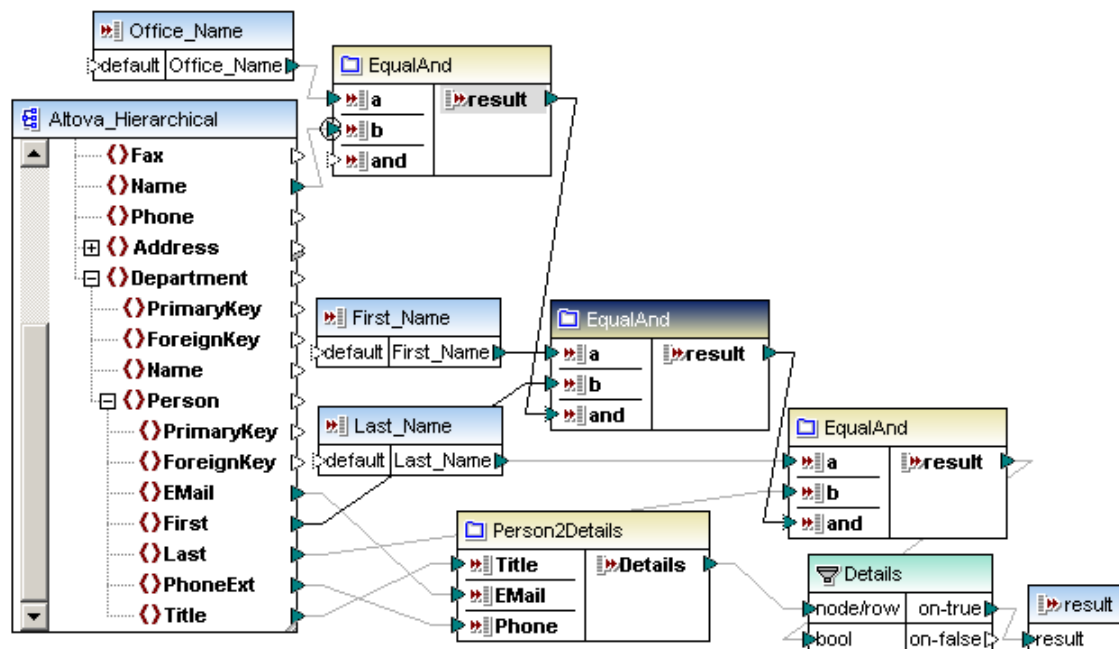
### Die LookupPerson-Komponente



Wenn Sie auf diese benutzerdefinierte Komponente doppelklicken, sehen Sie unterhalb davon die Komponenten, aus denen sie sich zusammensetzt. Diese Komponente hat folgende Funktionen:

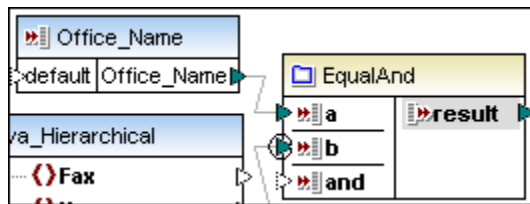
- Sie vergleicht mit Hilfe der **input**-Komponente und der benutzerdefinierten Komponente **EqualAnd** die Namen in Office\_Name, First\_Name und Last\_Name der Datei BranchOffices.xml mit den gleichnamigen Feldern der Datei Altova\_Hierarchical.xml.
- Sie kombiniert die Datenelemente Email, PhoneExt und Title mit Hilfe der benutzerdefinierten Funktion **Person2Details**.
- Sie übergibt die kombinierten Personendaten an die **output**-Komponente, wenn die Ergebnisse der vorherigen EqualAnd-Vergleiche alle "true" sind (d.h. es wird "true" an die filter-Komponente übergeben).

Eine benutzerdefinierte Funktion gibt immer einen Wert aus. Es kann sich dabei auch um einen leeren String handeln! Dies wäre der Fall, wenn der Boolesche Wert der filter-Komponente "false" ist. In diesem Fall würde nur ein leerer String anstelle der aus der Komponente Person2Details stammenden Daten ausgegeben.



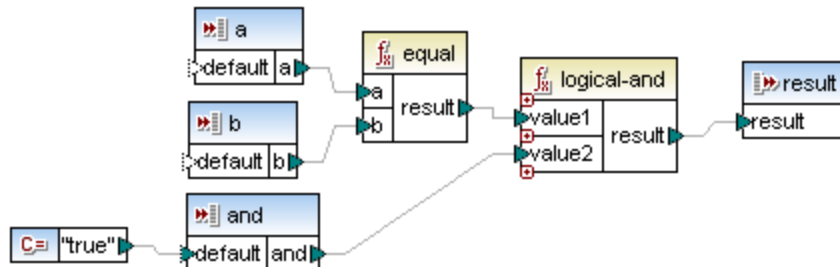
- Die drei **input**-Komponenten Office\_Name, First\_Name, Last\_Name erhalten ihre Daten aus der Datei BranchOffices.xml.
- Die **EqualAnd**-Komponente vergleicht zwei Werte und liefert einen **optionalen** Vergleichswert sowie einen Standardwert.
- Person2Details kombiniert drei Personenfelder und übergibt das Ergebnis an die Filterkomponente.

### Die EqualAnd-Komponente



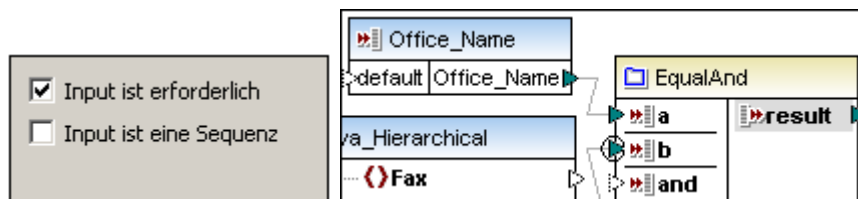
Wenn Sie auf diese benutzerdefinierte Komponente doppelklicken, werden die unten gezeigten Komponenten als ihre Bestandteile angezeigt. Diese Komponente:

- vergleicht zwei Eingabeparameter **a** und **b** und übergibt das Ergebnis an die logical-and-Komponente. Beachten Sie, dass der Parameter **b** als **Prioritätskontext** definiert wurde (rechtsklicken Sie auf das Symbol, um dies festzulegen). Damit stellen Sie sicher, dass die Personendaten des vom Eingabeparameter a bereitgestellten Büros zuerst verarbeitet werden.
- Die **Logical-and**-Komponente übergibt das Ergebnis des ersten Vergleichs mit einem **optionalen** Eingabeparameter "and".
- gibt den Booleschen Wert dieses Vergleichs an den Ausgabeparameter weiter.



### Optionale Parameter

Wenn Sie auf den "and" Parameter der oben gezeigten benutzerdefinierten Funktion "EqualAnd" doppelklicken, können Sie Parameter als optional definieren, indem Sie das Kontrollkästchen "Input ist erforderlich" deaktivieren.



Wenn "Input ist erforderlich" **deaktiviert** ist, dann:

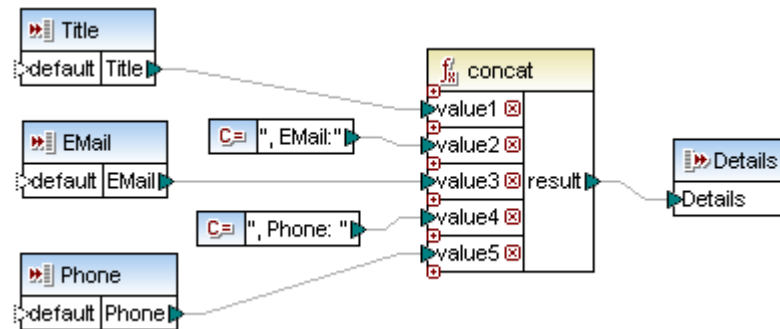
- wird für das Input-Symbol dieser benutzerdefinierten Funktion kein Mapping-Konnektor benötigt. So hat z.B. der **and**-Parameter der ersten EqualAnd-Funktion keinen Input-Konnektor. Das Input-Symbol erscheint strichliert, um dies visuell darzustellen.
- kann ein **default**-Wert angegeben werden, indem Sie eine Komponente innerhalb der benutzerdefinierten Funktion verbinden, z.B. mit Hilfe einer Konstanten-Komponente, die den Wert "true" enthält.
- hat ein Mapping von einem anderen Datenelement, das auf die optionale Eingabekomponente gemappt wird, Vorrang vor dem default-Wert. So erhält z.B. der "and"-Parameter der zweiten EqualAnd-Funktion Inputdaten vom "result"-Parameter der ersten benutzerdefinierten EqualAnd-Funktion.

### Die Person2Details-Komponente



Wenn Sie auf diese benutzerdefinierte Komponente doppelklicken, werden die unten gezeigten Bestandteilkomponenten angezeigt. Diese Komponente:

- verkettet die drei Inputs und übergibt den Ergebnis-String an den Ausgabeparameter.
- Wenn Sie auf einen Ausgabeparameter doppelklicken, können Sie den Parameternamen (Details) ändern und den Datentyp auswählen (String).



## 23.2 Hinzufügen benutzerdefinierter XSLT- und XQuery-Funktionen

In MapForce haben Sie die Möglichkeit, die installierten XSLT-Funktionsbibliotheken durch Ihre eigenen benutzerdefinierten Funktionen zu ergänzen. Wählen Sie dazu XSLT als Ausgabeformat, indem Sie auf das Symbol XSLT klicken oder wählen Sie **Ausgabe | XSLT 1.0**.

XSLT-Dateien werden als Bibliotheken angezeigt. Unterhalb des Bibliotheksnamens werden alle Named Templates als Funktionen angezeigt.

- Funktionen müssen als Named Templates deklariert werden, die der XSLT 1.0 Spezifikation in der XSLT-Datei entsprechen.
- Wenn die importierte XSLT-Datei andere XSLT-Dateien importiert oder beinhaltet, so werden auch diese XSLT-Dateien und Funktionen importiert.
- Jedes Named Template wird als Funktion unterhalb der einzelnen Bibliotheksnamen angezeigt.
- Die Anzahl der Input-Symbole, die gemappt werden können, hängt von der Anzahl der Parameter ab, die im Template Call verwendet werden; auch optionale Parameter werden unterstützt.
- Aktualisierungen der importierten XSLT-Dateien werden beim Programmstart oder bei Dateiänderungen durchgeführt.
- Namespaces werden unterstützt.

Bitte beachten Sie:

Stellen Sie beim Erstellen von Named Templates bitte sicher, dass die im Template verwendeten XPath-Anweisungen an den/die richtigen Namespace(s) gebunden sind. Sie können die Namespace Bindings des Mappings durch Klicken auf das Register XSLT anzeigen. Nähere Informationen dazu finden Sie auch unter der Dokumentation zur Implementierung von [XSLT 1.0](#).

### 23.2.1 Hinzufügen benutzerdefinierter Java .class- und .NET DLL-Funktionen

Sie können im Bibliotheksfenster kompilierte Java-Klassendateien sowie .NET Assembly-Dateien (einschließlich .NET 4.0 Assemblies) hinzufügen, die dann wie jede andere Funktion in MapForce verwendet werden können. Eine Vorschau der Mapping-Ausgabe dieser Java- und .NET-Funktionen kann im Ausgabefenster angezeigt werden. Außerdem stehen die Funktionen im generierten Code zur Verfügung.

Unterstützt:

- Kompilierte Java-Klassendateien (.class-Dateien) werden unterstützt, wenn als Ausgabesprache Java eingestellt ist.
- .NET Assembly-Dateien werden unterstützt, wenn als Ausgabesprache C# eingestellt ist. .NET Assemblies werden im Allgemeinen unterstützt unabhängig davon, was die Ausgangssprache war (C++ oder VB.NET), vorausgesetzt sie verwenden als Parameter und Rückgabetypen nur die Basisdatentypen aus der System Assembly.

Nicht unterstützt:

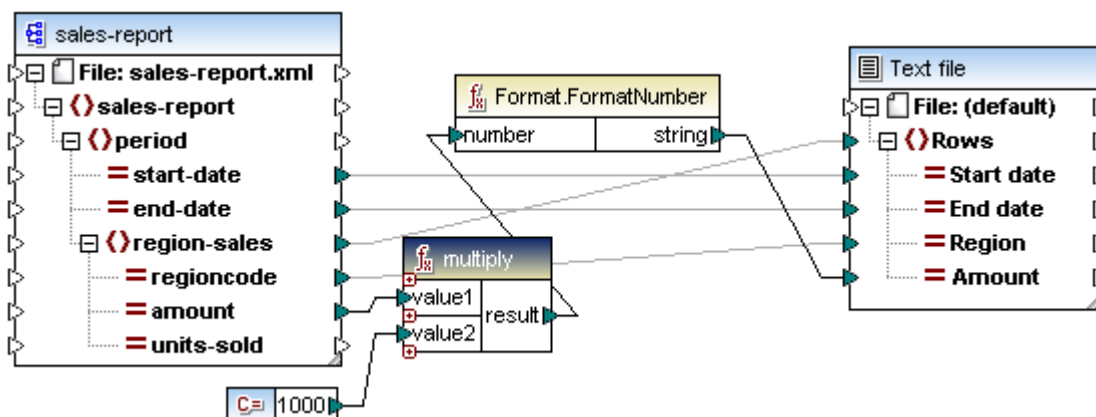
- Native C++ DLLs (oder andere DLLs, bei denen es sich nicht um eine .NET Assembly handelt)
- Verwendung von .class oder DLL Dateien/Funktionen, während als Ausgabesprache C++ eingestellt ist
- Verwendung von Java oder .NET Funktionen direkt in XSLT (es müsste eine benutzerdefinierte XSLT-Funktion geschrieben werden, die als Adapter fungiert)

Von C++-Funktionen kann keine Vorschau im Ausgabefenster angezeigt werden, doch steht die Ausgabe im generierten Code zur Verfügung.

**Um benutzerdefinierte Java- oder .NET-Funktionen hinzuzufügen, benötigen Sie:**

- die kompilierten Java-Klassen (.class) oder
- die .NET Assembly-Dateien (.dll)

In den Ordnern ...MapForceExamples\Java und ...MapForceExamples\C#\MapForce finden Sie Beispieldateien, in denen gezeigt wird, wie diese Funktionen verwendet werden. Beide Beispieldateien haben den Namen **FormatNumber.mfd**.

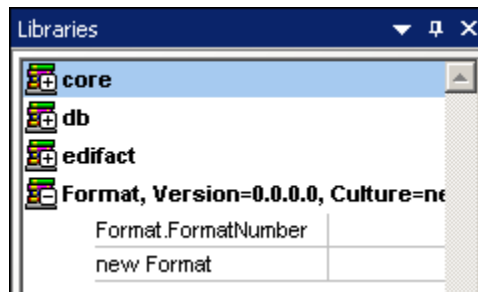


**So fügen Sie die .NET Assembly-Datei hinzu:**

1. Klicken Sie im Bibliotheksfenster auf die Schaltfläche "Bibliotheken hinzufügen/entfernen..." und im Ordner ...MapForceExamples\C#\Format\bin\Debug\ die Datei **Format.dll** aus.



Daraufhin wird eine Meldung angezeigt, dass eine neue Funktion hinzugefügt wurde. Die Funktion steht nun im Fenster "Bibliotheken" unter "Format" zur Verfügung.



**C# ausgewählt**

2. Öffnen Sie die Datei **FormatNumber.mfd** im Ordner ...\\MapForceExamples\\C# (siehe Abbildung oben).
3. Klicken Sie auf die Schaltfläche "Ausgabe", um das Ergebnis des Mappings zu sehen.

|    |                                       |
|----|---------------------------------------|
| 1  | Start date,End date,Region,Amount     |
| 2  | 2008-01-01,2008-01-31,CA,"110.400,00" |
| 3  | 2008-01-01,2008-01-31,MA,"75.300,00"  |
| 4  | 2008-02-01,2008-02-29,CA,"114.300,00" |
| 5  | 2008-02-01,2008-02-29,MA,"65.200,00"  |
| 6  | 2008-03-01,2008-03-31,CA,"134.200,00" |
| 7  | 2008-03-01,2008-03-31,MA,"86.100,00"  |
| 8  | 2008-04-01,2008-04-30,CA,"107.300,00" |
| 9  | 2008-04-01,2008-04-30,MA,"112.100,00" |
| 10 | 2008-05-01,2008-05-31,CA,"114.400,00" |
| 11 | 2008-05-01,2008-05-31,MA,"93.800,00"  |

Nähere Informationen zur Implementierung finden Sie unter: [Java- und .NET-Funktionen - Details](#).

### 23.2.2 Benutzerdefinierte XSLT 1.0-Funktionen

Die Dateien, die Sie für das unten gezeigte Beispiel benötigen, liegen im Verzeichnis **...MapForceExamples:**

- Name-splitter.xslt
- Name-splitter.xml (die XML-Instanzdatei für Customer.xsd)
- Customers.xsd
- CompletePO.xsd

Ein zusätzliches Beispiel für die Verwendung von Names Templates zum Summieren von Nodes finden Sie unter [Aggregationsfunktionen](#).

#### So fügen Sie eine benutzerdefinierte XSLT-Funktion hinzu:

1. Erstellen Sie eine XSLT-Datei, die die gewünschte Transformation/das gewünschte Ergebnis erzeugt.  
Im unten gezeigten Beispiel, **Name-splitter.xslt**, sehen Sie eine Named Template mit dem Namen **"tokenize"** mit einem einzigen Parameter "string". Die Vorlage verarbeitet einen Eingabe-String und trennt alle Großbuchstaben durch ein Leerzeichen voneinander.

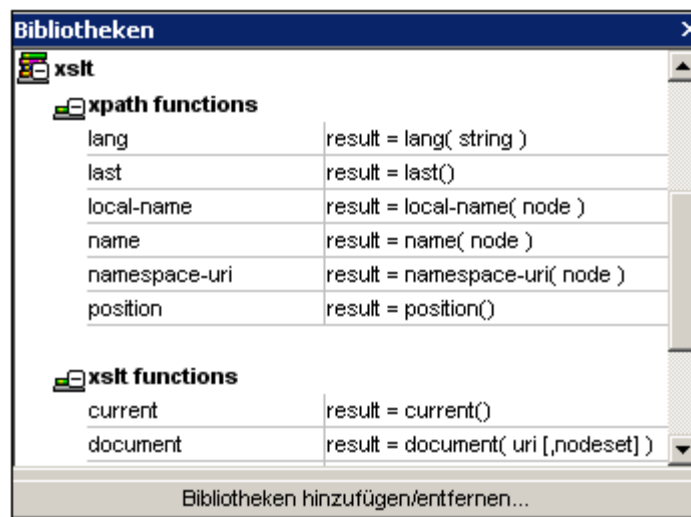


```

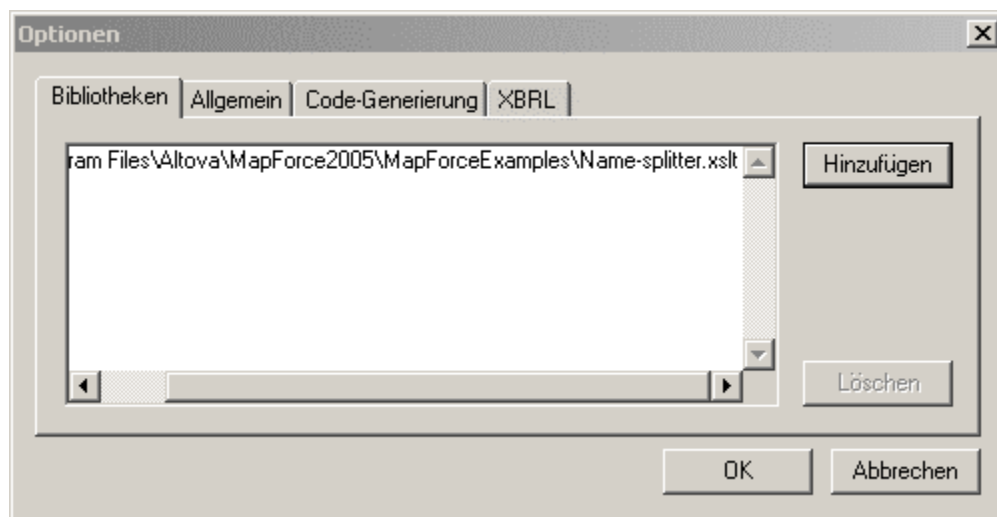
2 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
3 <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
4
5 <xsl:template match="*"
6 <xsl:for-each select="."
7 <xsl:call-template name="tokenize"
8 <xsl:with-param name="string" select="."
9 </xsl:call-template>
10 </xsl:for-each>
11 </xsl:template>
12
13 <xsl:template name="tokenize"
14 <xsl:param name="string" select="."
15 <xsl:variable name="caps" select="translate($string, '-abcdefghijklmnopqrstuvwxyz'
16 <xsl:variable name="capscount" select="string-length($caps)"
17 <xsl:variable name="token">

```

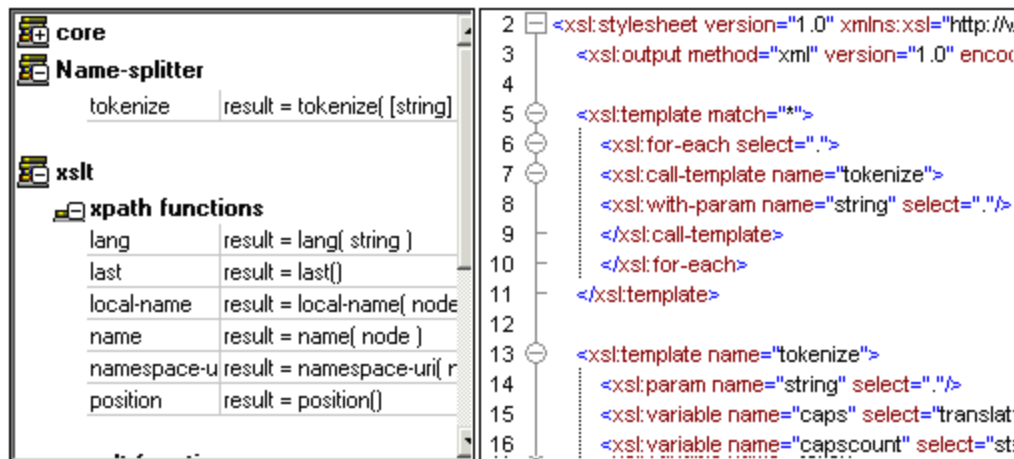
2. Klicken Sie auf die Schaltfläche **Bibliotheken hinzufügen/entfernen...** und im daraufhin angezeigten Dialogfeld auf "Hinzufügen".

**XSLT ausgewählt**

3. Wählen Sie die XSL- oder XSLT-Datei aus, die das named Template enthält, das als Funktion verwendet werden soll, in diesem Fall die Datei **Name-splitter.xslt**. Die XSLT-Datei wird im Bibliotheksfenster angezeigt.



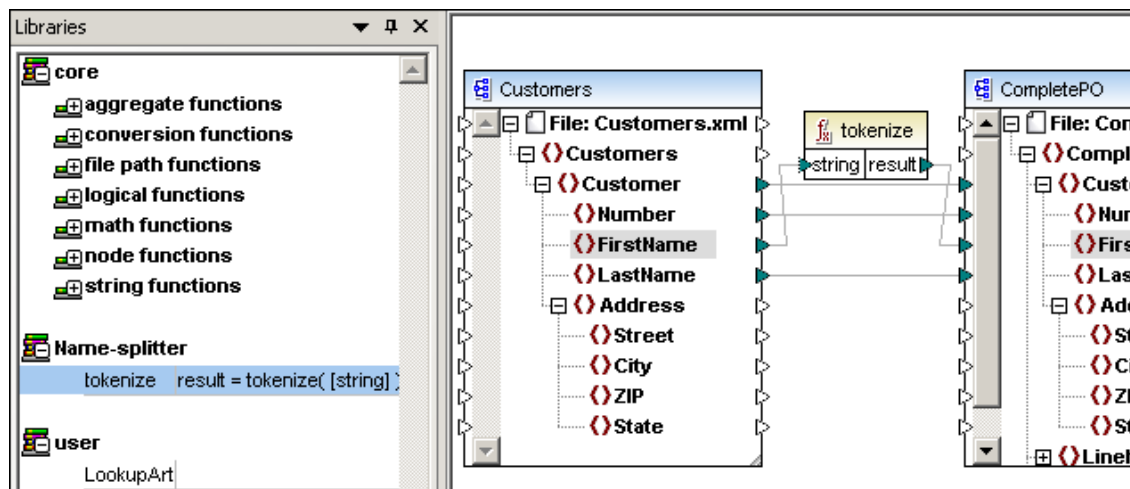
4. Klicken Sie auf **OK**, um die neue Funktion einzufügen.



### XSLT ausgewählt

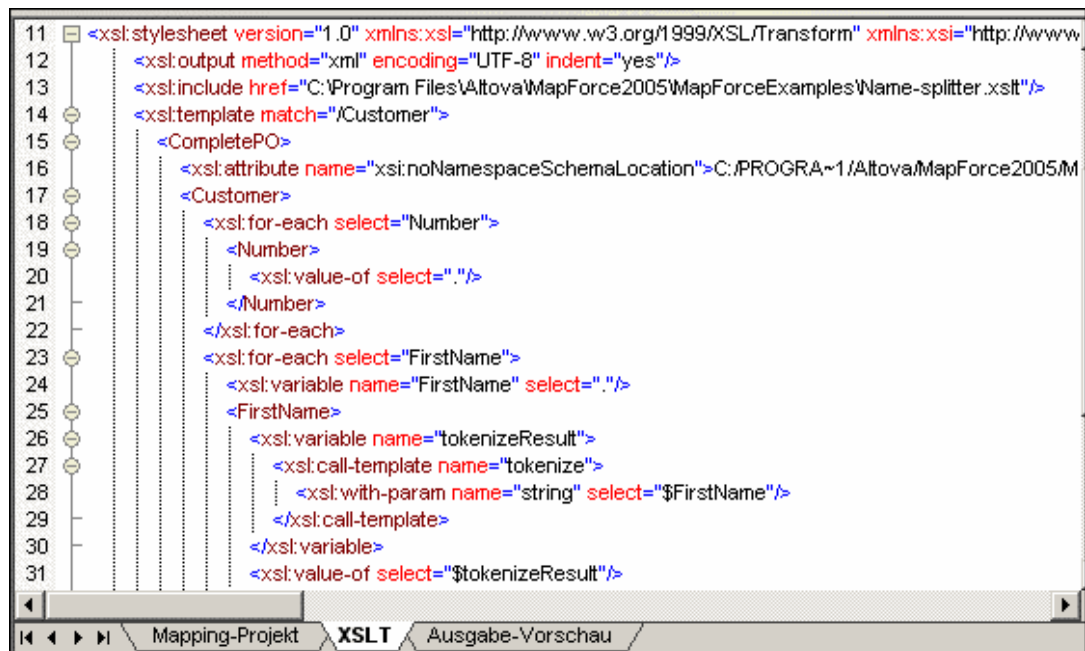
Die XSLT-Datei wird zusammen mit der/den als Named Template definierte(n) Funktion(en) im Bibliotheksfenster angezeigt. In diesem Fall sehen wir die Datei **Name-splitter** mit der unterhalb davon angezeigten Funktion **tokenize**.

5. Ziehen Sie die Funktion in das Mapping-Fenster, um sie in Ihrem aktuellen Mapping zu verwenden und mappen Sie die nötigen Datenelemente, wie in der unten stehenden Abbildung gezeigt.



### XSLT ausgewählt

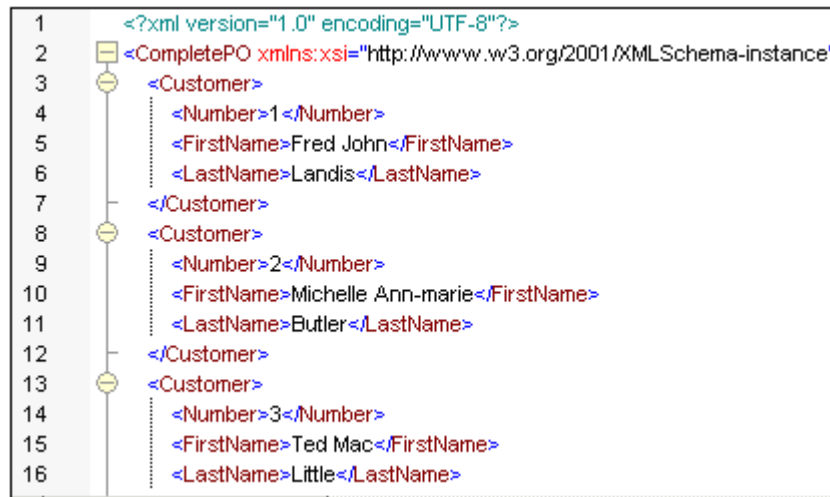
6. Klicken Sie auf das Register XSLT, um den erzeugten XSLT-Code zu sehen.



Bitte beachten Sie:

Sobald eine Named Template in einem Mapping verwendet wird, wird die XSLT-Datei, die den Namen des Named Template enthält, im generierten XSLT-Code **inkludiert** (**xsl:include href=...**) und mit dem Befehl **xsl:call-template** aufgerufen.

7. Klicken Sie auf die Schaltfläche **Ausgabe**, um das Ergebnis des Mappings anzuzeigen.



#### So löschen Sie benutzerdefinierte XSLT-Funktionen:

1. Klicken Sie auf die Schaltfläche **Bibliotheken hinzufügen/entfernen**.
2. Klicken Sie auf dem Bibliotheksregister auf den **Bibliotheksnamen** der gewünschten XSLT-Datei.
3. Klicken Sie auf die Schaltfläche **Löschen** und bestätigen Sie den Vorgang mit OK.

### 23.2.3 Benutzerdefinierte XSLT 2.0-Funktionen

In MapForce können Sie auch XSLT 2.0-Funktionen importieren, die in einem XSLT 2.0-Dokument in folgender Form vorkommen:

```
<xsl:function name="MyFunction">
```

Nähere Informationen dazu finden Sie in der Dokumentation zur Implementierung von [XSLT 2.0](#). Ein weiteres Beispiel zur Verwendung von Named Templates zum Summieren von Nodes finden Sie unter [Aggregationsfunktionen](#).

#### Datentypen in XPath 2.0

Wenn Ihr XML-Dokument ein XML-Schema referenziert und gemäß diesem Schema gültig ist, müssen Sie Datentypen, die nicht durch eine Operation implizit in den benötigten Datentyp konvertiert werden, explizit konstruieren oder "casten".

Im XPath 2.0 Datenmodell, das vom Altova XSLT 2.0-Prozessor verwendet wird, wird allen "**atomized**" Nodewerten aus dem XML-Dokument der Datentyp `xs:untypedAtomic` zugewiesen. Der Typ `xs:untypedAtomic` funktioniert gut mit impliziten Typkonvertierungen.

Beispiel:

- Der Ausdruck `xs:untypedAtomic("1") + 1` hat als Ergebnis den Wert 2, da der Wert `xdt:untypedAtomic` **implizit** vom Plus-Zeichen in `xs:double` geändert wird.
- Arithmetische Operatoren konvertieren Operanden implizit in `xs:double`.
- Operatoren zum Vergleich von Werten konvertieren Operanden vor dem Vergleich in `xs:string`.

### 23.2.4 Benutzerdefinierte XQuery-Funktionen

MapForce unterstützt den Import von XQuery-Bibliotheksmodulen.

Nähere Informationen dazu finden Sie in der Dokumentation zur Implementierung von [XQuery](#) .

### 23.2.5 Aggregatfunktionen - Aggregieren von Nodes in XSLT1 und 2

In diesem Abschnitt wird beschrieben, wie Sie mehrere Nodes eines XML-Instanzdokuments verarbeiten und die Ergebnisse als einen einzigen Wert auf das Ziel-Datenelement mappen können. Die in diesem Beispiel verwendeten Dateien liegen im Ordner **...MapForceExamples\Tutorial\** und bestehen aus:

|                          |                                                                         |
|--------------------------|-------------------------------------------------------------------------|
| Summing-nodes.mfd        | Mapping-Datei                                                           |
| input.xml                | XML-Input-Datei                                                         |
| input.xsd and output.xsd | Quell- und Ziel-Schemadateien                                           |
| Summing-nodes.xslt       | xslt file, die ein Named Template zum Summieren einzelner Nodes enthält |

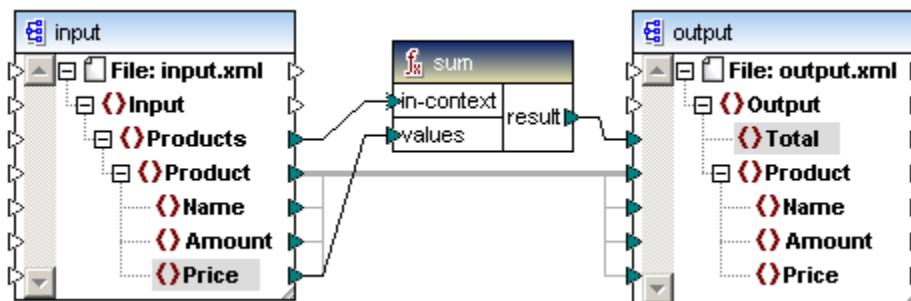
Es gibt zwei separate Methoden zum Erstellen und Verwenden von Aggregatfunktionen:

- Verwendung der Aggregatfunktionen in der **core-Bibliothek** im Bibliotheksfenster
- Verwenden eines **Named Template**.

#### Aggregatfunktionen - Bibliothek

Je nachdem, welche XSLT-Bibliothek Sie auswählen, XSLT 1 oder XSLT 2, stehen unterschiedliche Aggregatfunktionen in der Core-Bibliothek zur Verfügung. XSLT 1 unterstützt count und sum, während XSLT 2 avg, count, max, min, string-join und sum unterstützt.

Ziehen Sie die zu verwendende Aggregatfunktion aus der Bibliothek in das Mapping und verbinden Sie die Quell- und Zielkomponenten wie in der Abbildung unten gezeigt.



Nähere Informationen zu dieser Art von Aggregatfunktion finden Sie unter [Aggregatfunktionen](#).

#### Aggregatfunktionen - Named Templates

Im unten gezeigten Screenshot sehen Sie die **XML-Input-Datei**. Ziel dieses Beispiels ist, die Preisfelder beliebig vieler Produkte, in diesem Fall von Produkt A und B zu summieren.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <Input xmlns:xsi="http://www.w3.org/2001/XMLSchema
3 <Products>
4 <Product>
5 <Name>ProductA</Name>
6 <Amount>10</Amount>
7 <Price>5</Price>
8 </Product>
9 <Product>
10 <Name>ProductB</Name>
11 <Amount>5</Amount>
12 <Price>20</Price>
13 </Product>
14 </Products>
15 </Input>

```



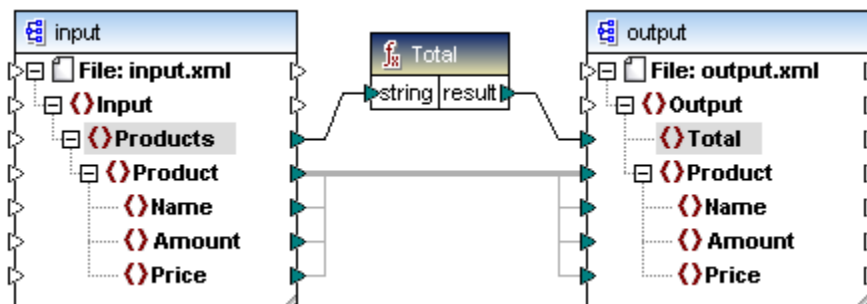
Im unten gezeigten Screenshot sehen Sie das XSLT Stylesheet, welches das Named Template "Total" und einen einzelnen Parameter "string" verwendet. Das Template arbeitet die XML-Input-Datei ab und summiert alle Werte, die mit dem XPath-Ausdruck **/Product/Price** im Dokument gefunden werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/19
 <xsl:output method="xml" version="1.0" encoding="UTF-8" i

 <xsl:template match="*">
 <xsl:for-each select=".">
 <xsl:call-template name="Total">
 <xsl:with-param name="string" select="."/>
 </xsl:call-template>
 </xsl:for-each>
 </xsl:template>

 <xsl:template name="Total">
 <xsl:param name="string"/>
 <xsl:value-of select="sum($string/Product/Price)"/>
 </xsl:template>
</xsl:stylesheet>
```

1. Klicken Sie auf die Schaltfläche **Bibliotheken hinzufügen/entfernen** und wählen Sie im Dialogfeld "Optionen" das Register "Bibliotheken".
2. Klicken Sie auf die Schaltfläche "Hinzufügen" und wählen Sie die Datei **Summing-nodes.xslt** aus dem Ordner **...\MapForceExamples\Tutorial**.
3. Ziehen Sie die Funktion "Total" aus der neu erstellten Bibliothek "Summing-nodes" ins Fenster und erstellen Sie Mappings wie unten gezeigt.



4. Klicken Sie auf die Schaltfläche "Ausgabe", um das Mapping-Ergebnis zu sehen.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Output xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNa
3 <Total>25</Total>
4 <Product>
5 <Name>ProductA</Name>
6 <Amount>10</Amount>
7 <Price>5</Price>
8 </Product>
9 <Product>
10 <Name>ProductB</Name>
11 <Amount>5</Amount>
12 <Price>20</Price>
13 </Product>
14 </Output>
15
```

Die beiden "Price"-Felder der beiden Produkten wurden addiert und in das Feld "Total" gesetzt.

**So summieren Sie die Nodes in XSLT 2.0:**

- Ändern Sie die Stylesheet-Deklaration in der Vorlage in ... version="2.0".

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsl:stylesheet version="2.0" xmlns:xs
3 <xsl:output method="xml" version="
```

## 23.3 Hinzufügen benutzerdefinierter Java, C# und C++-Funktionsbibliotheken

In MapForce können Sie ihre eigenen Funktionsbibliotheken für Java, C# und C++ erstellen und hinzufügen. Diese Funktionen können anschließend ähnlich wie vordefinierte Funktionen im grafischen Mapping verwendet werden.

Bibliotheken lassen sich durch Klicken auf die unterhalb des Bibliotheksfensters angezeigte Schaltfläche "Bibliotheken hinzufügen/entfernen..." hinzufügen. Sie können auch die Menüoption **Extras | Optionen | Hinzufügen** (auf dem Register "Bibliothek") verwenden. Bei Bibliotheken kann es sich um Java [.class](#)-Dateien, Visual Studio C# [.DLL](#)-Dateien sowie Dateien mit einer [.mff](#)-Erweiterung handeln.

Bitte beachten Sie bei **.MFF-Funktionen**:

Viele Mappings, bei denen diese Art von benutzerdefinierten Funktionen (.mff) verwendet wird, **können** über die Schaltfläche **Ausgabe**, also mit Hilfe des Built-In-Ausführungsprozessors **angezeigt werden**. Dies gilt für C#- und Java-Funktionen, jedoch nur für solche, die die nativen Typen der jeweiligen Sprache verwenden, nicht aber für Funktionen, die die generierten Altova-Klassen verwenden.

Alle diese Funktionen stehen natürlich bei der Codegenerierung zur Verfügung.

Benutzerdefinierte Funktionen, die in einem Mapping grafisch erstellt wurden, können und müssen nicht in einer \*.mff-Datei gespeichert werden bzw. dieser zugewiesen werden, da sie als Teil der Mapping-Datei gespeichert werden. Nähere Informationen zum Importieren und Verwalten benutzerdefinierter Funktionen finden Sie unter [Benutzerdefinierte Funktionen](#).

Zum Hinzufügen benutzerdefinierter \*.mff-Funktionen benötigen Sie:

- die MFF-Datei, die MapForce über die Schnittstellen zu den Funktionen informiert und
- wo die Implementierung für den generierten Code gefunden werden kann. Diese Implementierung ist eine Klasse in der entsprechenden Programmiersprache, die die in der mff-Datei definierten statischen Methoden enthält.

Eine einfache MFF-Datei für C# würde z.B. etwa folgendermaßen aussehen:

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xsi:noNamespaceSchemaLocation="mff.xsd" version="8" library="helloworld">
 <implementations>
 <implementation language="cs">
 <setting name="namespace" value="HelloWorldLibrary"/>
 <setting name="class" value="Greetings"/>
 <setting name="reference"
value="C:\HelloWorldLibrary\HelloWorldLibrary.dll"/>
 </implementation>
 </implementations>
 <group name="string functions">
 <component name="hello">
 <sources>
 <datapoint name="greeting_type" type="xs:boolean"/>
 </sources>
 <targets>
 <datapoint name="result" type="xs:string"/>
 </targets>
 <implementations>
 <implementation language="cs">
 <function name="HelloFunction"/>
 </implementation>
 </implementations>
 </component>
 </group>
</mapping>
```

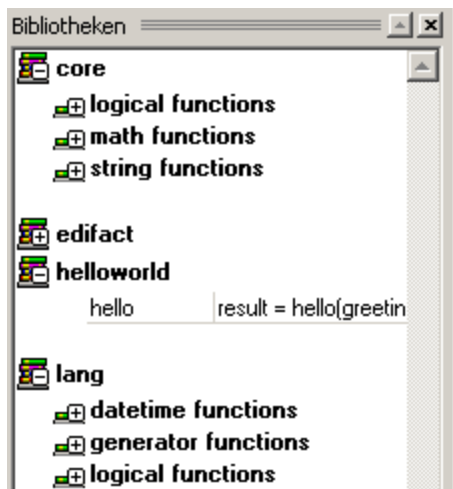
```

 </implementation>
 </implementations>
 <description>
 <short>result = hello(greeting_type)</short>
 <long>Returns a greeting sentence according to the given
greeting_type.</long>
 </description>
 </component>
</group>
</mapping>

```

Bitte beachten Sie: Die \*.mff-Bibliotheksdateien müssen gemäß der Schema-Datei im Verzeichnis ...\\MapForceLibraries gültig sein. Dieses Schema definiert die Konfiguration der benutzerdefinierten Bibliothek und dient nur zur internen Verwendung. Die Altova GmbH behält sich das Recht vor, dieses Dateiformat in neuen Releases zu ändern.

In der Abbildung unten sehen Sie, wie die MFF-Datei in MapForce angezeigt wird. Die neue Bibliothek "helloworld" erscheint als Eintrag in der Bibliotheksliste (in alphabetischer Reihenfolge sortiert) und enthält die String-Funktion "hello".



*Java ausgewählt*

MFF-Dateien können auch für mehrere Programmiersprachen angelegt werden. Jede zusätzliche Sprache muss daher ein zusätzliches <implementation>-Element enthalten. Die Einzelheiten zu den Implementierungselementen werden später in diesem Dokument erläutert.

Bitte beachten Sie, dass die exakte Reihenfolge, in der Funktionen vom generierten Mapping-Code aufgerufen werden, nicht definiert ist. MapForce kann berechnete Ergebnisse im Cache für die Wiederverwendung speichern oder Ausdrücke in jeder Reihenfolge auswerten. Es wird daher unbedingt empfohlen, nur Funktionen zu verwenden, die keine Nebeneffekte haben.

### 23.3.1 Konfigurieren der MFF-Datei

Im Folgenden sind die einzelnen Schritte beschrieben, wie man die MFF-Datei den eigenen Bedürfnissen entsprechend anpasst.

#### Der Bibliotheksname:

Der Bibliotheksname wird in der folgenden Zeile der MFF-Datei angezeigt. Laut Konvention wird der **Bibliotheksname** in **Kleinbuchstaben** geschrieben.

```
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xsi:noNamespaceSchemaLocation="mff.xsd"
 version="8" library="helloworld">
```

Die im Bibliotheksfenster angezeigte Überschrift wird den Namen "helloworld" haben. Diese Überschrift wird aber **nicht unbedingt** sofort, nachdem Sie im Einstellungsdialogfeld auf die Schaltfläche "Hinzufügen" geklickt haben, angezeigt. Bibliotheken werden erst dann angezeigt, wenn mindestens eine Komponente vorhanden ist, die eine Implementierung für die gerade ausgewählte Programmiersprache enthält.

Durch Hinzufügen oder Löschen der gewünschten Bibliotheksdatei (\*.mff) können Sie Bibliotheken und ihre Funktionen ein- und ausblenden.

#### So fügen Sie die neue MFF-Datei zum Bibliotheksfenster hinzu:

1. Klicken Sie auf die Schaltfläche "Bibliotheken hinzufügen/entfernen" und
2. dann auf die Schaltfläche "Hinzufügen".
3. Wählen Sie die benötigte MFF-Datei und klicken Sie auf die Schaltfläche "Öffnen", um sie zu laden.

Bitte beachten Sie:

Wenn Sie die Datei \*.mff im Ordner ...**MapForceLibraries** speichern, wird die Bibliothek beim Starten von MapForce automatisch geladen.

#### Implementierungselement für die Bibliothek "helloworld":

```
...
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:noNamespaceSchemaLocation="mff.
 xsd" version="8" library="helloworld">
 <implementations>
 <implementation language="cs">
 <setting name="namespace" value="HelloWorldLibrary"/>
 <setting name="class" value="Greetings"/>
 <setting name="reference"
value="C:\HelloWorldLibrary\HelloWorldLibrary.dll"/>
 </implementation>
 </implementations>
...

```

Für jede Sprache, die von der Bibliothek "helloworld" unterstützt werden soll, muss ein Implementierungselement hinzugefügt werden. Aufgrund der Einstellungen innerhalb jeder Implementierung kann der generierte Code die jeweilige in Java, C++ oder C# definierte Funktion abrufen.

Die spezifischen Einstellungen für die einzelnen Programmiersprachen werden unten beschrieben.

#### Java:

...

```

<implementation language="java">
 <setting name="package" value="com.hello.functions"/>
 <setting name="class" value="Hello"/>
</implementation>
...

```

Es ist wichtig, dass der generierte Code Ihre **hello.class**-Datei findet. Dazu muss sie im Classpath eingetragen sein. Der Standard-Classpath befindet sich in den Systemumgebungsvariablen.

Beachten Sie, dass es bei Standard-Java-Bibliotheken nur eine Klasse pro \*.mfh-Datei geben kann.

#### C#:

```

...
<implementation language="cs">
 <setting name="namespace" value="HelloWorldLibrary"/>
 <setting name="class" value="Hello"/>
 <setting name="reference" value="
C:\HelloWorldLibrary\HelloWorldLibrary.dll "/>
</implementation>
...

```

Es ist wichtig, dass der generierte Code bei C# unbedingt den hier definierten Namespace verwendet. Außerdem wird für C# auch der Pfad der **DLL** benötigt, die mit dem generierten Code verknüpft werden soll.

#### C++:

```

...
<implementation language="cpp">
 <setting name="namespace" value="helloworld"/>
 <setting name="class" value="Greetings"/>
 <setting name="path" value="C:\HelloWorldLibrary"/>
 <setting name="include" value="Greetings.h"/>
 <setting name="source" value="Greetings.cpp"/>
</implementation>
...

```

- **namespace** ist der Namespace, in dem Ihre **Greetings**-Klasse definiert werden wird. Er muss mit dem Bibliotheksnamen identisch sein.
- **path** ist der Pfad, unter dem die include-Dateien und die Quelldateien gespeichert sind.
- Bei Generierung von Code für ein Mapping werden die include-Datei und die Quelldatei in das Verzeichnis **Zielverzeichnis/Bibliotheksname** kopiert (Dieses Verzeichnis wird durch Auswahl der Menüoption **Datei | Code generieren in** und anschließende Auswahl des Verzeichnisses definiert) und in die Projektdatei inkludiert.

Alle von Ihnen definierten "include"-Dateien werden im generierten Algorithmus mit "include" inkludiert.

#### Hinzufügen einer Komponente:

Jede von Ihnen definierte Komponente wird sich in einer Funktionsgruppe befinden. Um beim helloworld-Beispiel zu bleiben:

```

...
<group name="string functions">
 <component name="hello">
 ...
 </component>
</group>
...

```

### 23.3.2 Definieren der Komponenten-Benutzerschnittstelle

Der unten stehende Code definiert, wie die Komponente angezeigt werden soll, wenn Sie in das Mapping-Fenster gezogen wird.

```
...
<component name="hello">
 <sources>
 <datapoint name="greeting_type" type="xs:boolean"/>
 </sources>
 <targets>
 <datapoint name="result" type="xs:string"/>
 </targets>
 <implementations>
 ...
 </implementations>
 <description>
 <short>result = hello(greeting_type)</short>
 <long>Returns a greeting sentence according to the given
greeting_type.</long>
 </description>
</component>
...
```

Die neue MapForce-Komponente:



#### Datapoints

Datapoints sind sozusagen die Eingabe- oder Ausgabeparameter einer Funktion. Der Datapoint-Parameter datatype definiert den Typ der Parameter/Return-Werte.

Bitte beachten Sie:

Für jede Funktion können nur ein **Ziel**-Datenpunkt, jedoch mehrere **Quell**-Datenpunkte definiert werden.

Beim Datentyp für die einzelnen Datenpunkte muss es sich um einen der folgenden handeln:

- einer der XML-Schematypen (z.B. xs:string, xs:integer, usw.)

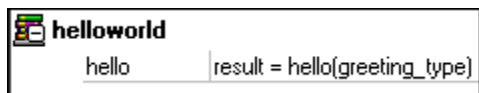
Diese Datentypen müssen den in Ihrer Java-, C++ oder C#-Bibliothek definierten Funktionsparametern entsprechen. Informationen über das Mappen von XML-Schemadatentypen auf Sprachtypen finden Sie in den Tabellen im Kapitel [Schreiben eigener Bibliotheken](#) unter der jeweiligen Programmiersprache.

Altova bietet für alle unterstützten Programmiersprachen Unterstützung für Schema SimpleTypes (date, time, duration, dateTime) als Klassen. Die Integration dieser Schema SimpleTypes in Ihrer Bibliothek wird an einer späteren Stelle in diesem Dokument beschrieben.

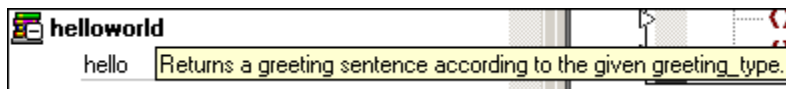
#### Funktionsbeschreibungen:

Funktionen sind im Bibliotheksfenster durch kurze und lange Beschreibungen ergänzt. Die Kurzbeschreibung wird immer rechts neben dem Funktionsnamen angezeigt, während die lange Beschreibung als Tooltip zu sehen ist, wenn Sie den Mauszeiger über die Kurzbeschreibung halten.

Kurzbeschreibung:



Lange Beschreibung:





### 23.3.3 Einzelheiten zur Funktionsimplementierung

Wir sind nun soweit, dass wir über das `<implementation>`-Element eine Verbindung zwischen einer Funktion im Bibliotheksfenster und der Funktion in der Java-, C#- oder C++-Klasse herstellen wollen.

Wie bereits erwähnt, kann eine Funktion mehrere "implementation"-Elemente haben - eines für jede der unterstützten Programmiersprachen.

```
...
<component name="hello">
...
 <implementations>
 <implementation language="cs">
 <function name="HelloFunction"/>
 </implementation>
 </implementations>
...
</component>
...
```

Eine Funktion kann den Namen "helloFunction" in Java oder "HelloFunctionResponse" in C++ haben. Aus diesem Grund müssen Sie für jede Programmiersprache separat einen Funktionsnamen definieren.

Eine Funktion für alle drei Programmiersprachen könnte folgendermaßen aussehen:

```
...
<component name="hello">
...
 <implementations>
 <implementation language="cs">
 <function name="HelloFunction"/>
 </implementation>
 <implementation language="java">
 <function name="helloFunction"/>
 </implementation>
 <implementation language="cpp">
 <function name="HelloFunctionResponse"/>
 </implementation>
 </implementations>
...
</component>
...
```

Der als Funktionsname verwendete Wert muss natürlich dem Namen der Methode in der Java-, C#- oder C++-Klasse exakt entsprechen.

### 23.3.4 Schreiben eigener Bibliotheken

Die Implementierung einer benutzerdefinierten Funktionsbibliothek ist eine Klasse in der entsprechenden Programmiersprache, die die in der mff-Datei definierten statischen Methoden für die einzelnen Funktionen enthält.

Bitte beachten Sie:

Wenn Sie ein Upgrade von einer MapForce Version vor Version 2010 durchführen, müssen Sie eventuell die in Ihren benutzerdefinierten Funktionen verwendeten Datentypen aktualisieren. Das aktuelle Mapping von XML-Schema-Datentypen auf native Datentypen finden Sie in den folgenden Abschnitten.

In den folgenden Abschnitten wird beschrieben, wie Sie Bibliotheken für eine bestimmte Programmiersprache erstellen:

[Erstellen einer Java-Bibliothek](#)

[Erstellen einer C#-Bibliothek](#)

[Erstellen einer C++-Bibliothek](#)

#### Erstellen einer Java-Bibliothek

##### So schreiben Sie eine Java-Bibliothek:

1. Erstellen Sie anhand des vorigen Beispiels eine neue Java-Klasse und geben Sie ihr den Namen "Hello".
2. Fügen Sie den Package-Namen hinzu, den Sie in den unten stehenden Zeilen angegeben haben:

```
....
<implementation language="java">
 <setting name="package" value="com.hello.functions"/>
 <setting name="class" value="Hello"/>
</implementation>
...
```

3. Wenn Sie spezielle XML-Schematypen (z.B. date, duration,...) benötigen, fügen Sie die Zeile  
import com.altova.types.\*;  
hinzu. Das genaue Mapping von XML-Schemadatentypen auf Java-Datentypen finden Sie in der unten stehenden Tabelle".

Wenn Sie Probleme haben, **com.altova.types** auf Ihrem Rechner zu finden, generieren Sie bitte Java-Code ohne benutzerdefinierte Funktionen; Sie werden die Klassen dann in dem von Ihnen angegebenen Verzeichnis finden.

4. Fügen Sie die Funktionen hinzu, die Sie in der MFF-Datei als public static definiert haben.

```
package com.hello.functions;

public class Hello {
 public static String HelloFunction (boolean greetingType) {
 if(greetingType)
 return "Hello World!";
 return "Hello User!";
 }
}
```

5. Kompilieren Sie die Java-Datei zu einer Klassen-Datei und fügen Sie diese Ihrem Classpath hinzu. Sie haben Ihre benutzerdefinierte Java-Bibliothek nun fertig erstellt.

### Datentyp-Mapping

| Schema-Typ         | Java-Typ                  |
|--------------------|---------------------------|
| anySimpleType      | String                    |
| anyAtomicType      | String                    |
| boolean            | boolean                   |
| string             | String                    |
| normalizedString   | String                    |
| token              | String                    |
| language           | String                    |
| NMTOKEN            | String                    |
| Name               | String                    |
| NCName             | String                    |
| ID                 | String                    |
| IDREF              | String                    |
| ENTITY             | String                    |
| untypedAtomic      | String                    |
| dateTime           | com.altova.types.DateTime |
| date               | com.altova.types.DateTime |
| time               | com.altova.types.DateTime |
| gYear              | com.altova.types.DateTime |
| gYearMonth         | com.altova.types.DateTime |
| gMonth             | com.altova.types.DateTime |
| gMonthDay          | com.altova.types.DateTime |
| gDay               | com.altova.types.DateTime |
| duration           | com.altova.types.Duration |
| base64Binary       | byte[]                    |
| hexBinary          | byte[]                    |
| anyURI             | String                    |
| QName              | javax.xml.namespace.QName |
| NOTATION           | String                    |
| double             | double                    |
| float              | double                    |
| decimal            | java.math.BigDecimal      |
| integer            | java.math.BigInteger      |
| nonPositiveInteger | java.math.BigInteger      |
| negativeInteger    | java.math.BigInteger      |
| long               | long                      |
| int                | int                       |
| short              | int                       |
| byte               | int                       |
| nonNegativeInteger | java.math.BigInteger      |
| positiveInteger    | java.math.BigInteger      |
| unsignedLong       | java.math.BigInteger      |
| unsignedInt        | long                      |
| unsignedShort      | long                      |
| unsignedByte       | long                      |
| dayTimeDuration    | com.altova.types.Duration |
| yearMonthDuration  | com.altova.types.Duration |
| NMTOKENS           | String                    |
| IDREFS             | String                    |
| ENTITIES           | String                    |

### Erstellen einer C#-Bibliothek

#### So schreiben Sie eine C#-Bibliothek:

1. Öffnen Sie ein neues Projekt in Visual Studio und erstellen Sie eine Klassen-Bibliothek.
2. Gehen Sie zu "add reference" und fügen Sie die Datei **Altova.dll** hinzu.

Wenn Sie Probleme haben, **Altova.dll** auf Ihrem Rechner zu finden, generieren Sie bitte C#-Code ohne benutzerdefinierte Funktionen; Sie werden die DLL dann in dem von Ihnen angegebenen Verzeichnis finden.

3. Wenn Sie spezielle XML-Schematypen (z.B. date, duration,...) benötigen, fügen Sie die Zeile

```
using Altova.Types;
```

hinzu. Das genaue Mapping von XML-Schemadatentypen auf C#-Datentypen finden Sie in der unten stehenden Tabelle.

4. Der Klassenname sollte mit dem von Ihnen definierten ident sein (hier "Greetings" ).

```
<implementation language="cs">
 <setting name="namespace" value="HelloWorldLibrary"/>
 <setting name="class" value="Greetings"/>
 <setting name="reference"
value="C:\HelloWorldLibrary\HelloWorldLibrary.dll"/>
</implementation>
```

5. Fügen Sie den Namespace hinzu. Verwenden Sie dafür denselben Wert, den Sie in den oben gezeigten mff Implementation Settings angegeben haben, also "HelloWorldLibrary".
6. Fügen Sie Ihre Funktionen als **public static** hinzu.

Der Beispielcode sollte folgendermaßen aussehen:

```
using System;
using Altova.Types;

namespace HelloWorldLibrary
{
 public class Greetings
 {
 public static string HelloFunction(bool GreetingType)
 {
 if(GreetingType)
 return "Hello World!";
 return "Hello User!";
 }
 }
}
```

7. Der letzte Schritt ist nun die Kompilierung des Codes.  
Der Pfad, unter dem die kompilierte DLL gespeichert ist, muss dem "reference"-Eintrag im "implementation"-Element entsprechen.

### Datentyp-Mapping

| Schema-Typ       | C# Typ |
|------------------|--------|
| anySimpleType    | string |
| anyAtomicType    | string |
| boolean          | bool   |
| string           | string |
| normalizedString | string |
| token            | string |

|                    |                       |
|--------------------|-----------------------|
| language           | string                |
| NMTOKEN            | string                |
| Name               | string                |
| NCName             | string                |
| ID                 | string                |
| IDREF              | string                |
| ENTITY             | string                |
| untypedAtomic      | string                |
| dateTime           | Altova.Types.DateTime |
| date               | Altova.Types.DateTime |
| time               | Altova.Types.DateTime |
| gYear              | Altova.Types.DateTime |
| gYearMonth         | Altova.Types.DateTime |
| gMonth             | Altova.Types.DateTime |
| gMonthDay          | Altova.Types.DateTime |
| gDay               | Altova.Types.DateTime |
| duration           | Altova.Types.Duration |
| base64Binary       | byte[]                |
| hexBinary          | byte[]                |
| anyURI             | string                |
| QName              | Altova.Types.QName    |
| NOTATION           | string                |
| double             | double                |
| float              | double                |
| decimal            | decimal               |
| integer            | decimal               |
| nonPositiveInteger | decimal               |
| negativeInteger    | decimal               |
| long               | long                  |
| int                | int                   |
| short              | int                   |
| byte               | int                   |
| nonNegativeInteger | decimal               |
| positiveInteger    | decimal               |
| unsignedLong       | ulong                 |
| unsignedInt        | ulong                 |
| unsignedShort      | ulong                 |
| unsignedByte       | ulong                 |
| dayTimeDuration    | Altova.Types.Duration |
| yearMonthDuration  | Altova.Types.Duration |
| NMTOKENS           | string                |
| IDREFS             | string                |
| ENTITIES           | string                |

### Erstellen einer C++-Bibliothek

#### So schreiben Sie eine C++-Bibliothek:

Erstellen Sie die **h**- und **cpp**-Dateien und speichern Sie sie unter genau demselben Namen und Pfad, den Sie im Implementation Element für die gesamte Bibliothek definiert haben.

"Header"-Datei:

1. Schreiben Sie "using namespace altova;"
2. Fügen Sie den von Ihnen angegebenen Namespace vom "implementation"-Element hinzu.
3. Fügen Sie die Klassen, die Sie im "implementation"-Element der MFF-Datei definiert haben, mit den in der MFF-Datei definierten statischen Funktionen hinzu.
4. Denken Sie daran, an den Beginn des Klassennamens "ALTOVA\_DECLSPECIFIER"

- zu schreiben, um sicherzustellen, dass Ihre Klassen korrekt kompiliert werden, egal ob Sie nun im anschließend generierten Code dynamic oder static linkage verwenden.
- Das genaue Mapping von XML-Schemadatentypen auf C++-Datentypen finden Sie in der unten stehenden Tabelle.

Die erzeugte **header-Datei** sollte folgendermaßen aussehen:

```
#ifndef HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED
#define HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED

#if _MSC_VER > 1000
 #pragma once
#endif // _MSC_VER > 1000

using namespace altova;

namespace helloworld {

class ALTOVA_DECLSPECIFIER Greetings
{
public:
 static string_type HelloFunctionResponse(bool greetingType);
};

} // namespace HelloWorldLibrary

#endif // HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED
```

Achten Sie in der **cpp**-Datei auf Folgendes:

- Die ersten Zeilen müssen die "includes" für **StdAfx.h**, und die Definitionen aus der **Altova**-Basisbibliothek sein. Kopieren Sie diese Zeilen bitte aus dem unten gezeigten Beispielcode hinein.
- Der Pfad **../Altova** ist für Ihre Quelldateien der richtige, da diese im erzeugten Code, den Sie unter Zielverzeichnis/Bibliotheksname finden werden, in ein separates Projekt kopiert werden.
- Als nächste Zeile folgt das "include" für Ihre oben erzeugte header-Datei.
- Fügen Sie die Implementierungen für Ihre Funktionen hinzu.
- Denken Sie bitte daran, dass sich die Implementierungen im richtigen Namespace, den Sie in der header-Datei angegeben haben, und im "implementations"-Element der MFF befinden müssen.

Die CPP-Beispieldatei würde folgendermaßen aussehen:

```
#include "StdAfx.h"
#include "../Altova/Altova.h"
#include "../Altova/AltovaException.h"
#include "../Altova/SchemaTypes.h"

#include "Greetings.h"

namespace helloworld {

 string_type Greetings::HelloFunctionResponse(bool greetingType)
 {
 if(greetingType)
 return _T("Hello World!");
 return _T("Hello User!");
 }

}
```

Im Gegensatz zu Java oder C# müssen Sie Ihre Quelldateien nicht kompilieren. Sie werden in den generierten Code kopiert und mit dem restlichen generierten Mapping-Code kompiliert.

#### C++-Kompilierfehler:

Wenn Sie in der unten gezeigten Zeile einen Kompilierfehler bekommen, fügen Sie den Pfad zur Datei msado15.DLL hinzu.

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

Sie müssen den Pfad zur Datei msado15.dll im Abschnitt directories (Verzeichnisse) Ihrer Visual-Studio-Umgebung einfügen.

1. Wählen Sie in VS in der Menüleiste: Tools / Options... aus.
2. Klicken Sie auf das Register "Directories".
3. Wählen Sie im Auswahlménü "Show directories for" den Eintrag "Include files".
4. Fügen Sie der Datei eine neue Zeile mit dem Pfadnamen hinzu;  
auf englischen Systemen lautet er normalerweise "C:\Program Files\Common Files\System\ADO"
5. Erstellen Sie die Datei neu und alles sollte nun ordnungsgemäß funktionieren.

#### Datentyp-Mapping

| Schema-Typ         | C++ Typ                |
|--------------------|------------------------|
| anySimpleType      | string_type            |
| anyAtomicType      | string_type            |
| boolean            | bool                   |
| string             | string_type            |
| normalizedString   | string_type            |
| token              | string_type            |
| language           | string_type            |
| NMTOKEN            | string_type            |
| Name               | string_type            |
| NCName             | string_type            |
| ID                 | string_type            |
| IDREF              | string_type            |
| ENTITY             | string_type            |
| untypedAtomic      | string_type            |
| dateTime           | altova::DateTime       |
| date               | altova::DateTime       |
| time               | altova::DateTime       |
| gYear              | altova::DateTime       |
| gYearMonth         | altova::DateTime       |
| gMonth             | altova::DateTime       |
| gMonthDay          | altova::DateTime       |
| gDay               | altova::DateTime       |
| duration           | altova::Duration       |
| base64Binary       | altova::mapforce::blob |
| hexBinary          | altova::mapforce::blob |
| anyURI             | string_type            |
| QName              | altova::QName          |
| NOTATION           | string_type            |
| double             | double                 |
| float              | double                 |
| decimal            | double                 |
| integer            | __int64                |
| nonPositiveInteger | __int64                |
| negativeInteger    | __int64                |

|                    |                  |
|--------------------|------------------|
| long               | __int64          |
| int                | int              |
| short              | int              |
| byte               | int              |
| nonNegativeInteger | unsigned __int64 |
| positiveInteger    | unsigned __int64 |
| unsignedLong       | unsigned __int64 |
| unsignedInt        | unsigned __int64 |
| unsignedShort      | unsigned __int64 |
| unsignedByte       | unsigned __int64 |
| dayTimeDuration    | altova::Duration |
| yearMonthDuration  | altova::Duration |
| NMTOKENS           | string_type      |
| IDREFS             | string_type      |
| ENTITIES           | string_type      |



## 23.4 Java- und .NET-Funktionen - Details

### Implementierungsdetails:

#### Hinzufügen von Bibliotheken:

- Java: `.class`-Dateien können zu MapForce hinzugefügt werden (`.jar`-Dateien werden nicht unterstützt)
- .NET: `.dll` Assembly-Dateien können zu MapForce hinzugefügt werden

**Warnung:** Alle von einem MapForce-Mapping aufgerufenen Funktionen sollten **"idempotent"** sein (Das bedeutet, sie sollten jedes Mal, wenn die Funktion mit denselben Input-Parametern aufgerufen wird, denselben Wert zurückgeben). Die genaue Reihenfolge und wie oft eine Funktion von MapForce aufgerufen wird, ist nicht definiert und kann sich jederzeit ändern.

#### Java-Funktionen - Konfiguration:

Wenn importierte Java-Klassendateien von anderen Klassendateien abhängig sind, stellen Sie sicher, dass Sie vor dem Start von MapForce die CLASSPATH-Umgebungsvariable anpassen. Zur CLASSPATH-Variablen müssen die übergeordneten Verzeichnisse aller abhängigen Pakete hinzugefügt werden.

Die importierten Klassendateien müssen nicht zur CLASSPATH-Variablen hinzugefügt werden, da der Built-In-Ausführungsprozessor - genau wie der generierte Java-Code - importierte Pakete automatisch zum Classpath bzw. zum ANT des Java-Prozessors hinzufügt.

### Unterstützung von Java-Funktionen

Klassen der obersten Ebene, statische Member-Klassen und nicht statische Member-Klassen werden unterstützt:

- `new <classname>(<arg1>, <arg2>, ...)`
- `<object>.new <member-class>(<arg1>, <arg2>, ...)`

Member-Funktionen und statische Funktionen werden unterstützt:

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

#### Typen

Unterstützte Verbindungen zwischen XML-Schema- und Java-Typen:

| Schema-Typ | Java-Typ |
|------------|----------|
| xs:string  | String   |
| xs:byte    | byte     |
| xs:short   | short    |
| xs:int     | int      |
| xs:long    | long     |
| xs:boolean | boolean  |
| xs:float   | float    |
| xs:double  | double   |

|            |                      |
|------------|----------------------|
| xs:decimal | java.math.BigDecimal |
| xs:integer | java.math.BigInteger |

Verbindungen können in beide Richtungen hergestellt werden. Andere Java-Typen werden nicht unterstützt.

Array-Typen werden nicht unterstützt. Methoden, die solche Parameter oder Rückgabewerte verwenden, werden ignoriert.

Object-Typen werden durch Aufruf ihres Konstruktors oder als Rückgabewert einer Methode unterstützt. Sie können auf andere Java-Methoden gemappt werden. Eine Bearbeitung des Objekts mit Hilfe von MapForce ist nicht möglich.

### Unterstützung von .NET-Funktionen

Klassen der obersten Ebene werden unterstützt:

- `new <classname>(<arg1>, <arg2>, ...)`

Member-Funktionen und statische Funktionen werden unterstützt:

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

### Typen:

Unterstützte Verbindungen zwischen XML-Schema- und .NET/C#-Typen:

| Schema-Typ       | .NET-Typ       | C#-Typ  |
|------------------|----------------|---------|
| xs:string        | System.String  | string  |
| xs:byte          | System.SByte   | sbyte   |
| xs:short         | System.Int16   | short   |
| xs:int           | System.Int32   | int     |
| xs:long          | System.Int64   | long    |
| xs:unsignedByte  | System.Byte    | byte    |
| xs:unsignedShort | System.UInt16  | ushort  |
| xs:unsignedInt   | System.UInt32  | uint    |
| xs:unsignedLong  | System.UInt64  | ulong   |
| xs:boolean       | System.Boolean | bool    |
| xs:float         | System.Single  | float   |
| xs:double        | System.Double  | double  |
| xs:decimal       | System.Decimal | decimal |

Verbindungen können in beide Richtungen hergestellt werden. Andere .NET/C#-Typen werden nicht unterstützt.

Array-Typen werden nicht unterstützt. Methoden, die solche Parameter oder Rückgabewerte verwenden, werden ignoriert.

Object-Typen werden durch Aufruf ihres Konstruktors oder als Rückgabewert einer Methode unterstützt. Sie können auf andere .NET-Methoden gemappt werden. Eine Bearbeitung des Objekts mit Hilfe von MapForce ist nicht möglich.

## 23.5 Referenz Funktionen

In diesem Abschnitt werden alle Funktionen beschrieben, die im Fenster "Bibliotheken" für die einzelnen unterstützten Sprachen XSLT1, XSLT2, XQuery, Java, C#, C++ sowie BUILTIN (den Built-in-Ausführungsprozessor) zur Verfügung stehen.

Derzeit stehen die folgenden Bibliotheken zur Verfügung:

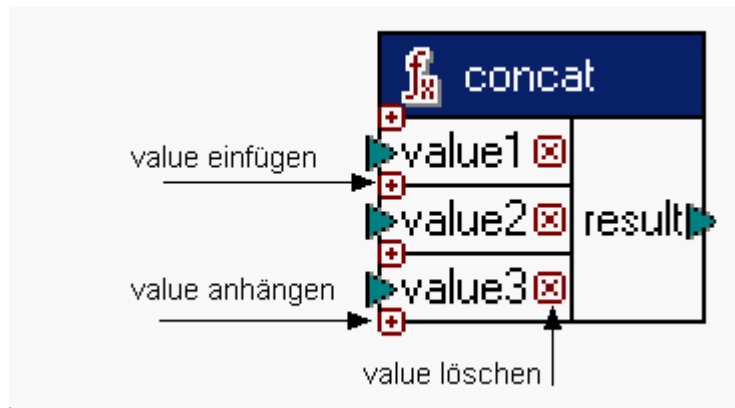
[core](#)  
[db](#)  
[edifact](#)  
[lang](#)  
[xbrl](#)  
[xlsx](#)  
[xpath2](#)  
[xslt](#)

### Erweiterbare Funktionen

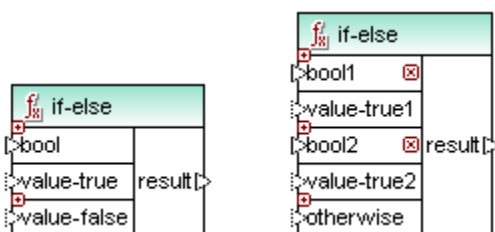
Eine Reihe von Funktionen aus den Funktionsbibliotheken kann erweitert werden, z.B: die Funktionen concat, "logical-and", "logical-or" und IF-ELSE. Die Parameter dieser Arten von Funktionen können je nach Belieben eingefügt/angehängt und gelöscht werden.

Wenn Sie auf das Plus-Symbol klicken, wird dieselbe Parameterart eingefügt oder angehängt. Wenn Sie auf das Häkchen klicken, wird der Parameter gelöscht.

Beachten Sie bitte. Wenn Sie einen Konnektor auf das Plus-Symbol ziehen, wird der Parameter automatisch eingefügt/angehängt und verbunden.



Die IF-Testparameter der [IF-Else](#) Funktion können auf dieselbe Art erweitert werden.



Wenn Sie den Mauszeiger über die Titelleiste der Funktion platzieren, erscheint ein Tooltip mit einer Beschreibung der Funktion.

Wenn Sie ihn über einen Parameter (jeden beliebigen Input- oder Ergebnisparameter) platzieren, wird der Datentyp des Arguments in einem Tooltipp angezeigt.

### 23.5.1 core

Die core-Bibliothek enthält die wichtigsten Funktionen für alle Sprachen. Die sequence-Funktionen stehen nicht zur Verfügung, wenn XSLT (XSLT 1.0) ausgewählt ist.

#### Core-Bibliothek

- [aggregates](#)
- [conversion functions](#)
- [file path functions](#)
- [generator functions](#)
- [logical functions](#)
- [math functions](#)
- [node functions](#)
- [sequence functions](#)
- [string functions](#)

#### aggregates (Aggregatfunktionen)

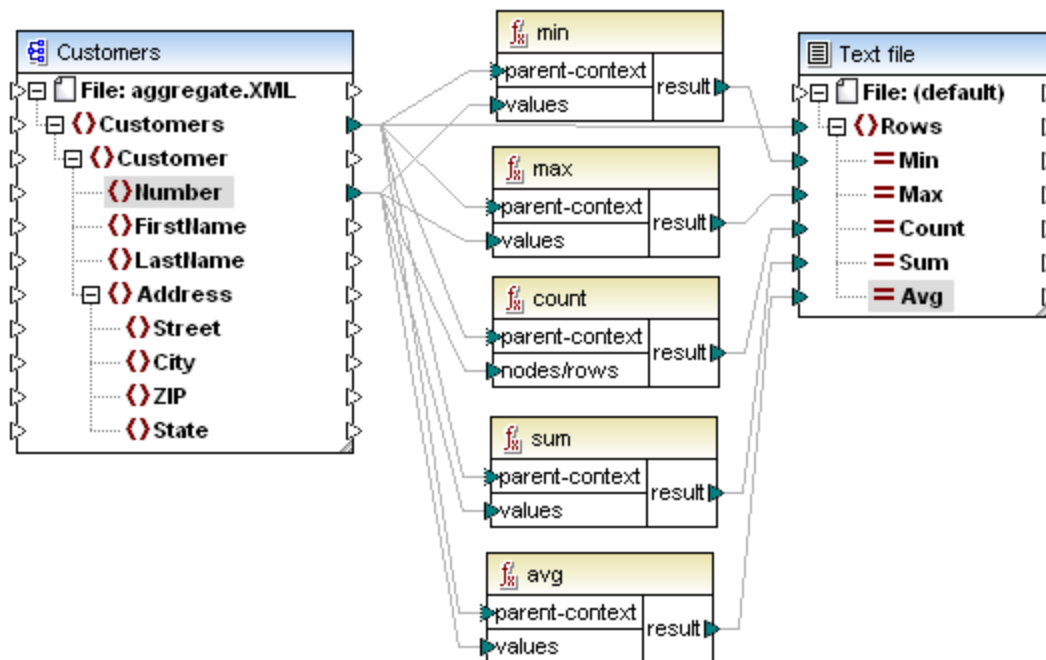
Mit Hilfe von Aggregatfunktionen werden Operationen an einer Gruppe oder Reihe von Input-Werten ausgeführt. Die Input-Daten für min, max, sum und avg werden für die Verarbeitung in den Datentyp **decimal** konvertiert.

- Die Input-Werte müssen mit dem **values** Parameter der Funktion verbunden werden.
- Der Kontextnode (Datenelement) muss mit dem **parent-context** Parameter verbunden werden, um den Standardkontext, aus dem die Input-Sequenz geholt wird, außer Kraft zu setzen. Das bedeutet auch, dass der parent-context-Parameter optional ist.
- Das Ergebnis der Funktion (**result**) wird mit dem entsprechenden Zieldatenelement verbunden.

Das nachfolgende Mapping finden Sie unter dem Namen Aggregates.mfd im Ordner ....\Tutorial. Es zeigt, wie diese Funktionen verwendet werden.

Aggregatfunktionen haben zwei Input-Elemente.

- **values** (nodes/rows) ist mit dem Quellelement verbunden, aus dem die Daten stammen, in diesem Fall "Number".
- **parent-context** ist mit dem Element verbunden, an dem iteriert werden soll, in diesem Fall allen Kunden ("Customer"). Der Parameter ist allerdings optional.



Bei der Input-Instanz handelt es sich in diesem Fall um eine XML-Datei mit den folgenden Daten:

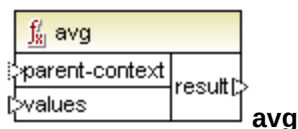
Comment edited with XMLSPY v2004 U (<http://www.xmlspy.com>) by M

| Customers                                                        |        |                   |
|------------------------------------------------------------------|--------|-------------------|
| <div>xmlns:xsi</div> <div>http://www.w3.org/2001/XMLSchema</div> |        |                   |
| <div>xsi:noNamespace...</div> <div>Customers.xsd</div>           |        |                   |
| Customer (4)                                                     |        |                   |
|                                                                  | Number | Firstname         |
| 1                                                                | 2      | FredJohn          |
| 2                                                                | 4      | MichelleAnn-marie |
| 3                                                                | 6      | TedMac            |
| 4                                                                | 8      | AnnLong           |

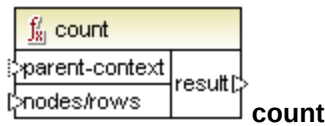
- Bei den Quelldaten, die an das Element "values" übergeben werden, handelt es sich um die Zahlenfolge 2,4,6,8.
- Bei der Output-Komponente handelt es sich in diesem Fall um eine einfache Textdatei. Wenn Sie für das obige Mapping auf das Register "Ausgabe" klicken, erhalten Sie das folgende Ergebnis:

|   |            |
|---|------------|
| 1 | 2,8,4,20,5 |
| 2 |            |

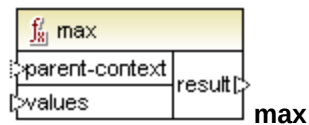
min=2, max=8, count=4, sum=20 and avg=5.



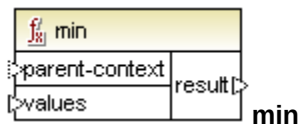
Gibt den Durchschnittswert aller Werte in der Input-Sequenz zurück. Der Durchschnittswert einer leeren Gruppe ist eine leere Gruppe. Steht in XSLT1 nicht zur Verfügung.



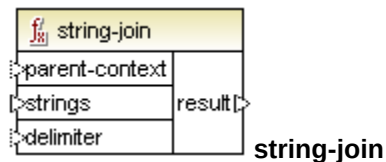
Gibt die Anzahl der einzelnen Datenelemente zurück, aus denen die Input-Sequenz besteht. Die Anzahl einer leeren Gruppe ist Null. Eingeschränkte Funktionalität in XSLT1.



Gibt den Maximalwert aller Werte in der Input-Sequenz zurück. Der Maximalwert einer leeren Gruppe ist eine leere Gruppe. Steht in XSLT1 nicht zur Verfügung.



Gibt den Minimalwert alle Werte in der Input-Sequenz zurück. Der Minimalwert einer leeren Gruppe ist eine leere Gruppe. Steht in XSLT1 nicht zur Verfügung.

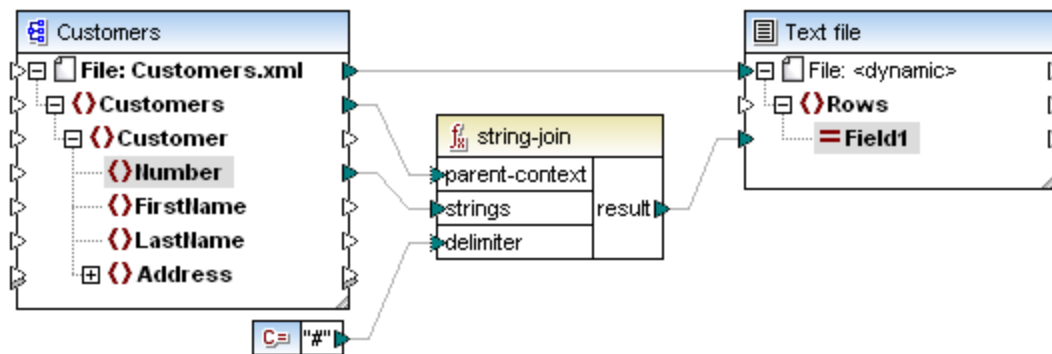


Verkettet alle Werte der Input-Sequenz zu einem String, der durch ein beliebiges von Ihnen ausgewähltes Trennzeichen getrennt ist. Der **string-join** Wert einer leeren Gruppe ist ein leere String. Steht in XSLT1 nicht zur Verfügung.

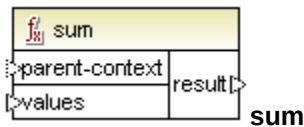
Das Beispiel unten enthält vier separate Kundennummern 2 4 6 und 8. Das Konstantenzeichen liefert als Trennzeichen eine Raute "#".

Ergebnis = 2#4#6#8





Wenn Sie kein Trennzeichen definieren, ist die Standardeinstellung ein leerer String, d.h. keinerlei Trennzeichen. Ergebnis = 2468.



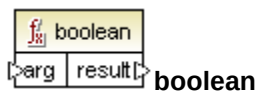
Gibt die arithmetische Summe aller Werte in der Input-Sequenz zurück. Die Summe einer leeren Gruppe ist Null. Steht in XSLT1 nicht zur Verfügung.

### conversion functions (Konvertierungsfunktionen)

Zur Unterstützung expliziter Datentypkonvertierungen stehen in der Bibliothek "conversion functions" die Konvertierungsfunktionen "boolean", "number" und "string" zur Verfügung. Beachten Sie, dass MapForce notwendige Konvertierungen in den meisten Fällen automatisch durchführt und dass diese Funktionen nur in speziellen Fällen verwendet werden müssen.

Input-Parameter = **arg/value**

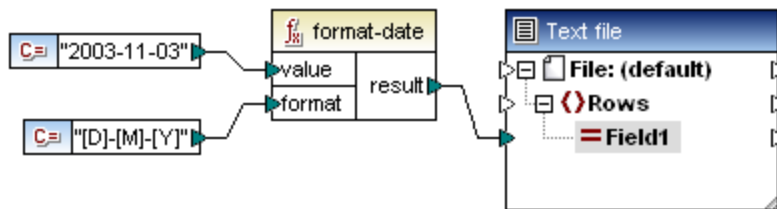
Output-Parameter = **result**



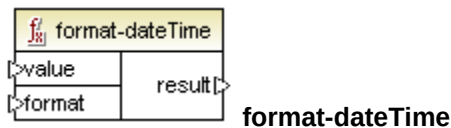
Konvertiert einen numerischen Input-Wert in einen Booleschen (und konvertiert einen String in einen numerischen Wert - true in 1). Wert, z.B. 0 in "false" oder 1 in "true". Dieser Wert kann anschließend in logischen Funktionen (gleich, größer als usw.), Filtern oder if-else-Funktionen verwendet werden.



Konvertiert einen xs:date Eingabewert in einen String.



Ergebnis: is 3-11-2003



Konvertiert einen dateTime-Datentyp in einen String.

| Argument | Beschreibung                                                         |
|----------|----------------------------------------------------------------------|
| value    | Der zu formatierenden dateTime-Wert                                  |
| format   | Ein Formatstring, der definiert, wie dateTime formatiert werden soll |

Anmerkung:

Wenn die Ausgabe der Funktion (also result) mit einem Node verbunden ist, dessen Typ nicht "string" ist, kann die Formatierung verloren gehen, da der Wert in den Zieltyp konvertiert wird. Diese automatische Konvertierung kann durch Deaktivieren des Kontrollkästchens "[Zielwerte in Zieltypen konvertieren](#)" in den Komponenteneinstellungen der Zielkomponente abgeschaltet werden.

### Formatierungsstring

Das Argument "format" besteht aus einem String, der so genannte Variablen-Marker enthält, die innerhalb von eckigen Klammern stehen. Die Zeichen außerhalb der eckigen Klammern sind Literalzeichen, die in das Ergebnis kopiert werden. Wenn das Ergebnis eckige Klammern als Literalzeichen enthalten soll, so sollten diese eckigen Klammern verdoppelt werden.

Jeder Variablen-Marker besteht aus einem Komponenten-Specifier, der angibt, welche Komponente des Datums oder der Uhrzeit angezeigt werden soll, einem optionalen Formatierungs-Modifier, einem weiteren optionalen Darstellungs-Modifier und einem optionalen Breiten-Modifier. Falls diesen Modifiern steht, falls vorhanden, ein Komma.

```
format := (literal | argument)*
argument := [component(format)?(presentation)?(width)?]
width := , min-width ("-" max-width)?
```

Die Komponenten sind:

| Specifier | Beschreibung          | Standarddarstellung |
|-----------|-----------------------|---------------------|
| Y         | Jahr (absoluter Wert) | vier Stellen (2010) |
| M         | Monat des Jahres      | 1-12                |
| D         | Tag des Monats        | 1-31                |
| d         | Tag des Jahres        | 1-366               |

|   |                                                                   |                                             |
|---|-------------------------------------------------------------------|---------------------------------------------|
| F | Wochentag                                                         | Name des Tages (sprachabhängig)             |
| W | Woche des Jahres                                                  | 1-53                                        |
| w | Woche des Monats                                                  | 1-5                                         |
| H | Stunde (24 Stunden)                                               | 0-23                                        |
| h | Stunde (12 Stunden)                                               | 1-12                                        |
| P | A.M. oder P.M.                                                    | alphabetisch (sprachabhängig)               |
| m | Minuten in der Stunde                                             | 00-59                                       |
| s | Sekunden in der Minute                                            | 00-59                                       |
| f | Sekundenbruchteile                                                | numerisch, eine Dezimalstelle               |
| Z | Zeitzone gerechnet als Zeitabstand zur UTC                        | +08:00 oder PST mit alphabetischem Modifier |
| z | Zeitzone gerechnet als Zeitabstand zur GMT (Westeuropäische Zeit) | GMT+n                                       |

Der Darstellungs-Modifier:

| Specifizier | Beschreibung                                  | Beispiel                          |
|-------------|-----------------------------------------------|-----------------------------------|
| o           | Bezeichnet die Nummerierung in Ordnungszahlen | [1o] bezeichnet 1., 2., 3., 4.... |

Der Formatierungs-Modifier:

| Zeichen | Beschreibung                                                                        | Beispiel         |
|---------|-------------------------------------------------------------------------------------|------------------|
| 1       | Numerisches Dezimalformat, wobei keine Null vorangestellt ist: 1, 2, 3, ...         | 1, 2, 3          |
| 01      | Dezimalformat, zwei Stellen: 01, 02, 03, ...                                        | 01, 02, 03       |
| i       | Römische Ziffern                                                                    | i, ii, iii, iv   |
| I       | Römische Zahlen, Großbuchstaben                                                     | I, II, III, IV   |
| N       | Name der Komponente, Großbuchstaben                                                 | MONTAG, DIENSTAG |
| n       | Name der Komponente, Kleinbuchstaben                                                | monday, tuesday  |
| Nn      | Name der Komponente, erster Buchstabe ist ein Großbuchstabe                         | Montag, Dienstag |
| W       | Ausgeschriebene Zahl, Großbuchstaben (nicht für das Parsen)                         | EINS, ZWEI, DREI |
| w       | Ausgeschriebene Zahl, in Kleinbuchstaben (nicht für das Parsen)                     | eins, zwei, drei |
| Ww      | Ausgeschriebene Zahl, erster Buchstabe ist ein Großbuchstabe (nicht für das Parsen) | Eins, Zwei, Drei |

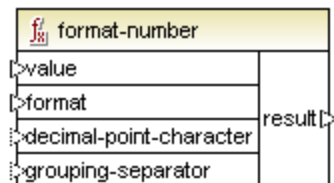
1) Die Modifier N, n und Nn unterstützen nur die folgenden Komponenten: M, d, D.

Vor dem Modifizier "width", steht, falls vorhanden, ein Komma. Der Modifizier hat die folgende Form:

, min-width ("-" max-width)?

#### Unterstützte Beispiele

| DateTime            | Formatierungsstring                                | Ergebnis                                      |
|---------------------|----------------------------------------------------|-----------------------------------------------|
| 2003-11-03T00:00:00 | [D]/[M]/[Y]                                        | 3/11/2003                                     |
| 2003-11-03T00:00:00 | [Y]-[M,2]-[D,2]                                    | 2003-11-03                                    |
| 2003-11-03T00:00:00 | [Y]-[M,2]-[D,2]                                    | 2003-11-03                                    |
| 2003-11-03T00:00:00 | [Y]-[M,2]-[D,2] [H,2]:[m]:[s]                      | 2003-11-03 00:00:00                           |
| 2010-06-02T08:02    | [Y] [MNn] [D01] [F,3-3] [d]<br>[H]:[m]:[s].[f]     | 2010 June 02 Wed 153<br>8:02:12.054           |
| 2010-06-02T08:02    | [Y] [MNn] [D01] [F,3-3] [d]<br>[H]:[m]:[s].[f] [z] | 2010 June 02 Wed 153<br>8:02:12.054 GMT+02:00 |
| 2010-06-02T08:02    | [Y] [MNn] [D1] [F] [H]:[m]:[s].[f]<br>[Z]          | 2010 June 2 Wednesday<br>8:02:12.054 +02:00   |
| 2010-06-02T08:02    | [Y] [MNn] [D] [F,3-3]<br>[H01]:[m]:[s]             | 2010 June 2 Wed<br>08:02:12                   |



**format-number**

Verfügbar für XSLT 1.0, XSLT 2.0, Java, C#, C++ und den Built-In-Ausführungsprozessor.

| Argument             | Beschreibung                                                                                                                             |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| value                | Die zu formatierende Zahl                                                                                                                |
| format               | Ein Formatierungsstring, der festlegt, wie die Zahl formatiert werden soll                                                               |
| decimal-point-format | Das Zeichen, das für das Dezimalzeichen, also im Deutschen das Komma, verwendet wird. Standardeinstellung ist der Punkt '.' (optional)   |
| grouping-separator   | Das Trennzeichen, das zur Trennung von Tausenderstellen verwendet werden soll. Standardmäßig ist das Zeichen ',' eingestellt (optional). |

Anmerkung:

Wenn die Ausgabe der Funktion (also result) mit einem Node verbunden ist, dessen Typ nicht "string" ist, kann die Formatierung verloren gehen, da der Wert in den Zieltyp konvertiert wird. Diese automatische Konvertierung kann durch Deaktivieren des Kontrollkästchens "[Zielwerte in Zieltypen konvertieren](#)" in den Komponenteneinstellungen der Zielkomponente abgeschaltet werden.

### format

Ein Formatierungsstring legt fest, wie eine Zahl formatiert werden soll.

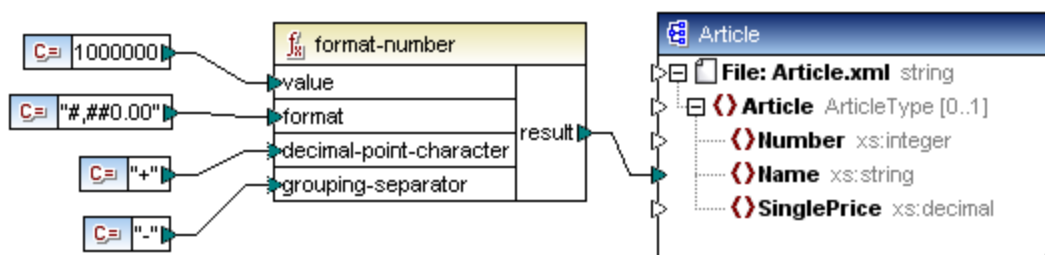
Format:

```
format := subformat (;subformat)?
subformat := (prefix)? integer (.fraction)? (suffix)?
prefix := any characters except special characters
suffix := any characters except special characters
integer := (#)* (0)* (allowing ',' to appear)
fraction := (0)* (#)* (allowing ',' to appear)
```

Das erste *subformat* dient zum Formatieren positiver Zahlen, das zweite zum Formatieren negativer Zahlen. Wenn nur ein *subformat* definiert ist, so wird dasselbe Unterformat auch für negative Zahlen verwendet, wobei aber vor dem *prefix* ein Minuszeichen steht.

| Sonderzeichen      | Standard | Beschreibung                                                                                                                                          |
|--------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| zero-digit         | 0        | Eine Stelle, die immer an dieser Stelle im Ergebnis angezeigt wird                                                                                    |
| digit              | #        | An dieser Stelle erscheint im Ergebnisstring eine Ziffer, es sei denn es handelt sich um eine nicht benötigte vorangestellte oder nachgestellte Null. |
| decimal-point      | .        | Trennt den Ganzzahlenbereich in der Zahl von den Kommastellen.                                                                                        |
| grouping-seperator | ,        | Trennt Zifferngruppen (Tausender-Trennzeichen).                                                                                                       |
| percent-sign       | %        | Multipliziert die Zahl mit 100 und zeigt sie als Prozentwert an                                                                                       |
| per-mille          | ‰        | Multipliziert die Zahl mit 1000 und zeigt Sie als Promille an                                                                                         |

Die Zeichen, die als Dezimalpunktzeichen und als Tausender-Trennzeichen verwendet werden, sind immer die Zeichen "." bzw. ",". Durch Mappen von Konstanten auf diese Nodes kann dies allerdings in der formatierten Ausgabe geändert werden.



Das Ergebnis oben gezeigten Zahlenformatierungsfunktion.

- Der Dezimalpunkt wird in ein "+" Zeichen geändert.

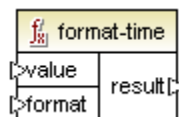
- Das Tausender-Trennzeichen wurde in ein "-" Zeichen geändert.

```
<Article xsi:noNamespaceSchemaLocation="
 <Name>1-000-000+00</Name>
</Article>
```

### Rounding

Die für diese Funktion verwendete Rundungsmethode ist, Aufrunden ab der Hälfte, d.h. wenn der Wert hinter dem Komma > oder gleich, 0,5 ist, so wird aufgerundet. Wenn der Wert kleiner als 0,5 ist, wird abgerundet.

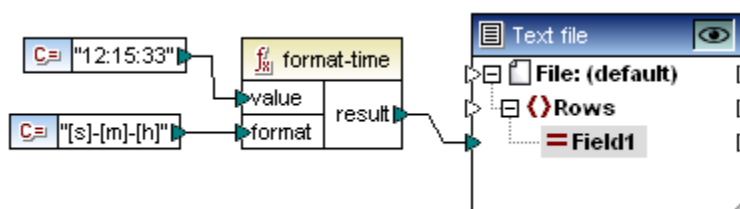
| Zahl    | Formatierungsstring | Ergebnis     |
|---------|---------------------|--------------|
| 1234.5  | #,##0.00            | 1,234.50     |
| 123.456 | #,##0.00            | 123.46       |
| 1000000 | #,##0.00            | 1,000,000.00 |
| -59     | #,##0.00            | -59.00       |
| 1234    | ###0.0###           | 1234.0       |
| 1234.5  | ###0.0###           | 1234.5       |
| .00025  | ###0.0###           | 0.0003       |
| .00035  | ###0.0###           | 0.0004       |
| 0.25    | #00%                | 25%          |
| 0.736   | #00%                | 74%          |
| 1       | #00%                | 100%         |
| -42     | #00%                | -4200%       |
| -3.12   | #.00;(#.00)         | (3.12)       |
| -3.12   | #.00;#.00CR         | 3.12CR       |



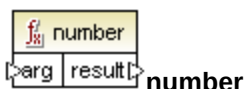
**format-time**

Konvertiert einen xs:time Eingabewert in einen String.

z.B.



Ergebnis: 33-15-12



Konvertiert einen Input String in eine Zahl. Konvertiert einen Booleschen Input auch in eine Zahl.



Verfügbar für Java, C#, C++ und den Built-In-Ausführungsprozessor.

Konvertiert einen String in ein Datum und ignoriert die Uhrzeitkomponente. Für diese Funktion wird als Grundlage die Funktion [parse-dateTime](#) verwendet, wobei die Uhrzeitkomponente ignoriert wird. Das Ergebnis ist vom Typ xs:date.



Verfügbar für Java, C#, C++ und den Built-In-Ausführungsprozessor.

Konvertiert einen String in einen dateTime-Wert.

| Argument | Beschreibung                                                        |
|----------|---------------------------------------------------------------------|
| value    | Der String, der Datum und Uhrzeit enthält.                          |
| format   | Der Bildstring, der definiert, wie der Wert derzeit formatiert wird |

Die Komponenten sind:

| Parameter | Beschreibung                      | Standarddarstellung                         |
|-----------|-----------------------------------|---------------------------------------------|
| Y         | Jahr (absoluter Wert)             | vier Stellen (2010)                         |
| M         | Monat des Jahres                  | 1-12                                        |
| D         | Tag des Monats                    | 1-31                                        |
| d         | Tag des Jahres                    | 1-366                                       |
| H         | Stunde (24 Stunden)               | 0-23                                        |
| h         | Stunde (12 Stunden)               | 1-12                                        |
| P         | A.M. oder P.M.                    | alphabetisch (sprachabhängig)               |
| m         | Minuten in Stunde                 | 00-59                                       |
| s         | Sekunden in Minute                | 00-59                                       |
| f         | Sekundenbruchteile                | numerisch, eine Dezimalstelle               |
| Z         | Zeitzone, als Zeitabstand von UTC | +08:00 oder PST mit alphabetischem Modifier |
| z         | Zeitzone als Zeitabstand von GMT  | GMT+n                                       |

Der Formatierungsparameter:

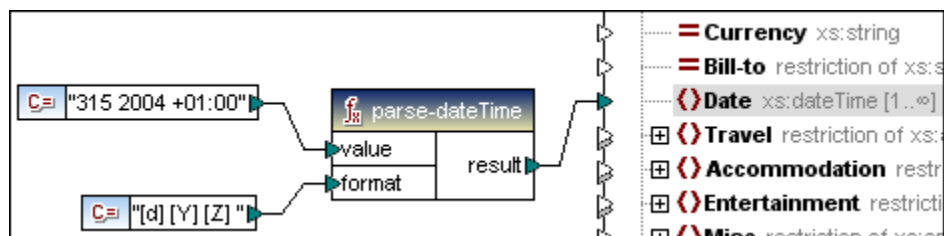
| Zeichen | Beschreibung                                                     | Beispiel        |
|---------|------------------------------------------------------------------|-----------------|
| 1       | Numerisches Dezimalformat ohne vorangestellte Null: 1, 2, 3, ... | 1, 2, 3         |
| 01      | Dezimalformat, zwei Ziffern: 01, 02, 03, ...                     | 01, 02, 03      |
| N       | Name der Komponente, Großbuchstaben                              | MONDAY, TUESDAY |
| n       | Name der Komponente, Kleinbuchstaben                             | monday, tuesday |
| Nn      | Name der Komponente, beginnend mit einem Großbuchstaben          | Monday, Tuesday |

1) Die Parameter **N**, **n** und **Nn** unterstützen nur die Komponente M (Monat).

Beispiele:

| Datumsstring                         | Bildstring                               | Ergebnis                      |
|--------------------------------------|------------------------------------------|-------------------------------|
| 21-03-2002 16:21:12.492<br>GMT+02:00 | [D]-[M]-[Y]<br>[H]:[m]:[s].[f] [z]       | 2002-03-21T16:21:12.492+02:00 |
| 315 2004 +01:00                      | [d] [Y] [Z]                              | 2004-11-10T00:00:00+01:00     |
| 1.December.10 03:2:39 p.m.<br>+01:00 | [D].[MNn].[Y,2-2]<br>[h]:[m]:[s] [P] [Z] | 2010-12-01T15:02:39+01:00     |
| 20110620                             | [Y,4-4][M,2-2][D,2-2]                    | 2011-06-20T00:00:00           |

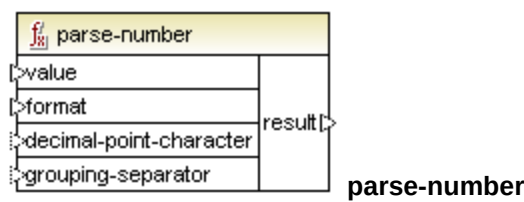
Beispiel in MapForce:



Ergebnis:

```
<expense-item xmlns="http://my-company.com/nam
C:/DOCUME~1/MYDOCU~1/Altova/MapForce2020
 <Date>2004-11-10T00:00:00+01:00</Date>
</expense-item>
```





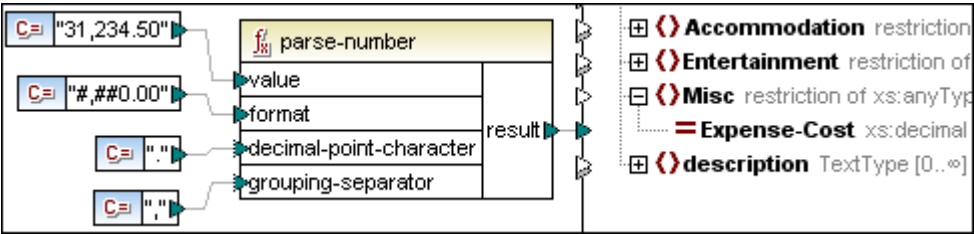
Verfügbar für Java, C#, C++ und den Built-In-Ausführungsprozessor.

Konvertiert einen Input-String in eine Dezimalzahl.

| Argument                | Beschreibung                                                                                                               |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------|
| value                   | Der zu parsende/in eine Zahl zu konvertierende String                                                                      |
| format                  | Ein Formatierungsstring, der angibt, wie die Zahl derzeit formatiert ist (optional). Die Standardeinstellung ist "#,##0.#" |
| decimal-point-character | Das als Dezimaltrennzeichen zu verwendende Zeichen. Standardeinstellung ist das Zeichen '.' (optional)                     |
| grouping-separator      | Das für separate Zahlengruppen zu verwendenden Trennzeichen. Standardeinstellung ist das Zeichen ',' (optional)            |

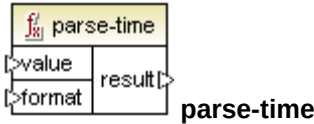
Der in parse-number verwendete **Formatierungsstring** ist derselbe, der auch in [format-number](#) verwendet wird.

Beispiel in MapForce:



Ergebnis:

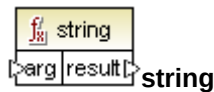
```
<expense-item xmlns="http://my-company.com/ham
C:/DOCUME~1/MYDOCU~1/Altova/MapForce2011
 <Misc MiscExpense-Cost="31234.5"/>
</expense-item>
```



Verfügbar für Java, C#, C++ und den Built-In-Ausführungsprozessor.

Konvertiert einen String in eine Uhrzeit und ignoriert die Datumskomponente. Für diese Funktion wird als Grundlage die Funktion [parse-dateTime](#) verwendet, wobei die

Datumskomponente ignoriert wird. Das Ergebnis ist vom Typ xs:date.

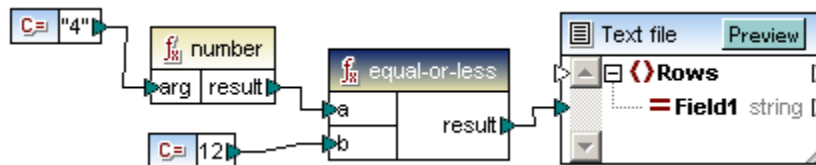


Konvertiert einen Input-Wert in einen String. Sie können mit dieser Funktion auch den Textinhalt eines Node abrufen.

Wenn es sich beim Input-Node um einen complexType handelt, so werden alle untergeordneten Nodes ebenfalls als einzelner String ausgegeben.

### Vergleich unterschiedlicher Input Node Typen

Wenn es sich bei den Input Nodes um unterschiedliche Typen handelt, z.B. Integer und String, können Sie mit Hilfe der Konvertierungsfunktionen einen String-Vergleich oder einen numerischen Vergleich erzwingen.



Im Beispiel oben hat die erste Konstante den Typ string und enthält den String "4".

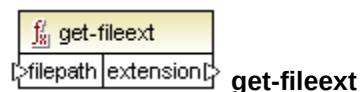
Die zweite Konstante enthält die numerische Konstante 12. Damit die beiden Werte explizit verglichen werden können, müssen sie denselben Typ haben.

Durch Hinzufügen einer **number** Funktion zur ersten Konstante wird die string-Konstante in den numerischen Wert 4 konvertiert. Das Ergebnis der Vergleiche ist dann "true".

Beachten Sie: Wenn die number-Funktion nicht verwendet würde und 4 direkt mit einem Parameter verbunden würde, würde es zu einem String-Vergleich kommen, wobei das Ergebnis "false" wäre.

### file path functions (Dateipfadfunktionen)

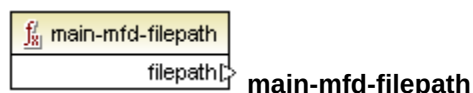
Mit Hilfe von Dateipfadfunktionen können Sie Dateipfaddaten, also Ordner, Dateinamen und Erweiterungen direkt aufrufen und für die weitere Verarbeitung in Ihren Mappings bearbeiten. Diese Funktionen können in allen Sprachen, die von MapForce unterstützt werden, verwendet werden.



Gibt die Erweiterung des Dateipfads einschließlich des Punktzeichens "." zurück.  
z.B. gibt 'c:\data\Sample.mfd' die Erweiterung '.mfd' zurück.



Gibt den Ordernamen des Dateipfads einschließlich des Schrägstrichs oder umgekehrten Schrägstrichs zurück.  
z.B. gibt 'c:/data/Sample.mfd' 'c:/data/' zurück.

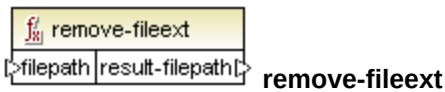


Gibt den vollständigen Pfad der mfd-Datei, die das Hauptmapping enthält, zurück. Wenn die mfd-Datei derzeit noch nicht gespeichert ist, wird ein leerer String zurückgegeben.



Wenn die Funktion im Hauptmapping aufgerufen wird, gibt die Funktion dasselbe wie die Funktion main-mfd-filepath zurück, also den vollständigen Dateipfad der mfd-Datei, die das Haupt-Mapping enthält. Wenn die mfd-Datei derzeit noch nicht gespeichert ist, wird ein leerer String zurückgegeben.

Wenn die Funktion von einer **benutzerdefinierten Funktion** aus aufgerufen wird, die von einer mfd-Datei **importiert** wird, gibt sie den vollständigen Pfad der importierten mfd-Datei zurück, die die **Definition** der benutzerdefinierten Funktion enthält.



Entfernt die Erweiterung des Dateipfads einschließlich des Punkts.  
z.B. gibt 'c:/data/Sample.mfd' 'c:/data/Sample' zurück.



Entfernt das Verzeichnis des Dateipfads einschließlich des nachgestellten Schrägstrichs oder umgekehrten Schrägstrichs.  
z.B. gibt 'c:/data/Sample.mfd' 'Sample.mfd' zurück.



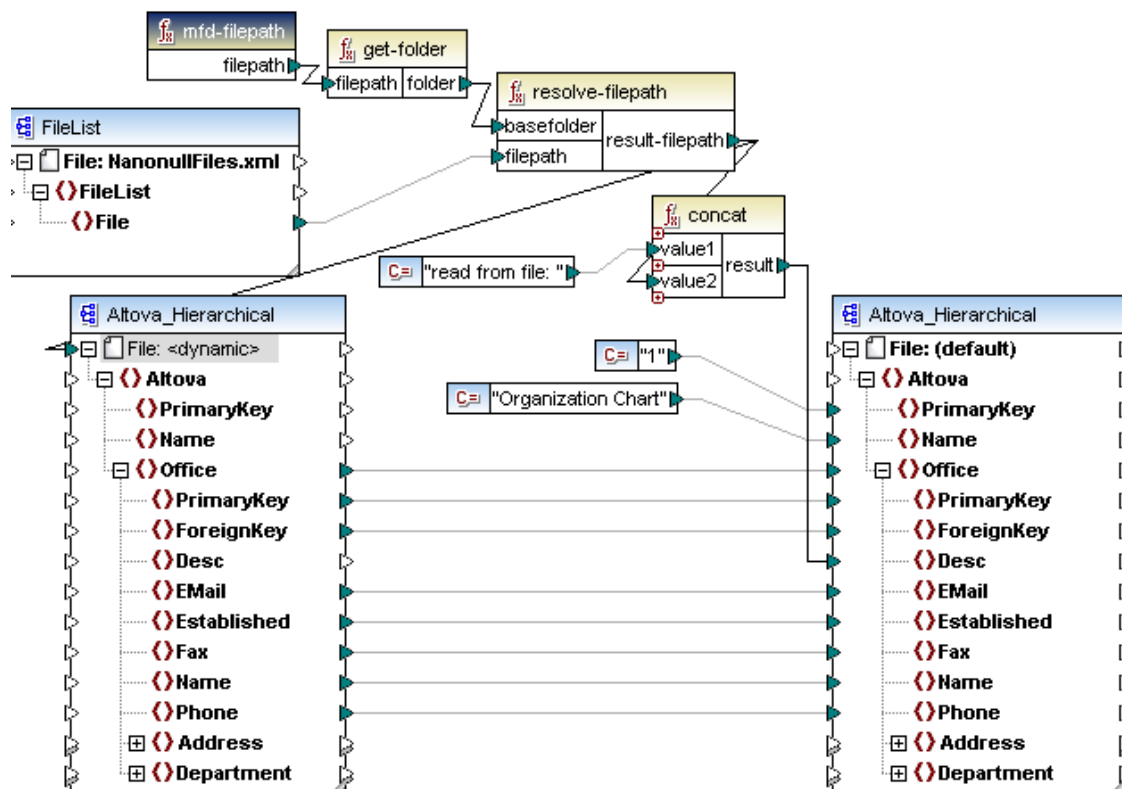
Ersetzt die durch den filepath-Parameter bereitgestellte Erweiterung des Dateipfads durch die Erweiterung, die durch die Verbindung zum extension-Parameter bereitgestellt wird.

z.B. gibt c:/data/Sample.mfd' als Input-Dateipfad und '.mfp' als die Erweiterung 'c:/data/Sample.mfp' zurück.



Löst einen relativen Dateipfad zu einem relativen oder absoluten Basisordner auf. Die Funktion unterstützt '.' (aktuelles Verzeichnis) und '..' (übergeordnetes Verzeichnis).

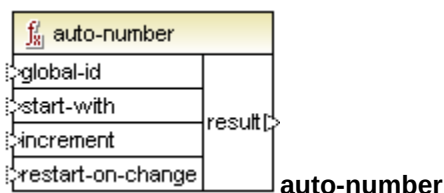
Ein Beispiel dazu finden Sie im Mapping **MergeMultipleFiles\_List.mfd** im Ordner ... \MapForceExamples.



### generator functions (Generierungsfunktionen)

Mit Hilfe der **auto-number**-Funktion können Sie abhängig von den verschiedenen Parametern, die Sie definieren, Ganzzahlen in den Ziel-Nodes einer Komponente generieren.

Stellen Sie sicher, dass der result-Konnektor (der auto-number-Funktion) **direkt** mit einem Ziel-Node verbunden ist. Die exakte Reihenfolge, in der Funktionen vom generierten Mapping-Code aufgerufen werden, ist nicht definiert. MapForce kann die berechneten Ergebnisse zur Wiederverwendung im Cache aufbewahren oder Ausdrücke in jeder beliebigen Reihenfolge auswerten. Seien Sie daher bei der Verwendung der auto-number-Funktion besonders vorsichtig.



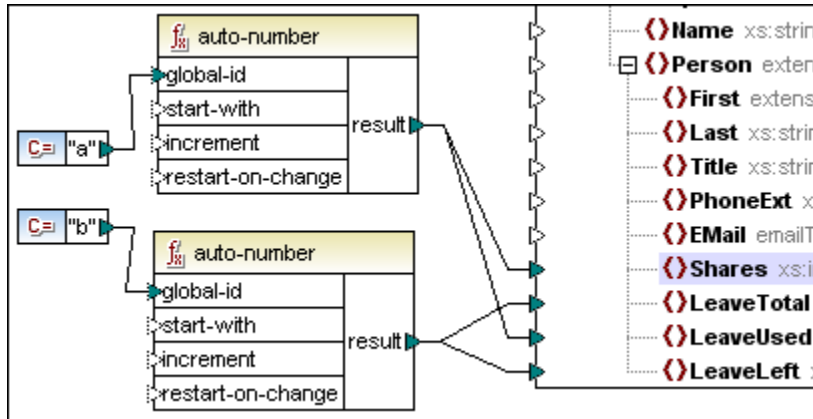
Das Ergebnis ist ein Wert, der bei **start\_with** beginnt und durch **increment** inkrementiert wird. Die Standardwerte beginnen mit 1 (start-with=1) und werden um 1 (increase=1) erhöht. Beide Parameter können negativ sein.

### global-id

Mit Hilfe dieses Parameters können Sie die Zahlensequenz Ausgabe von zwei separaten mit einer einzigen Zielkomponente verbundenen auto-number-Funktionen synchronisieren.

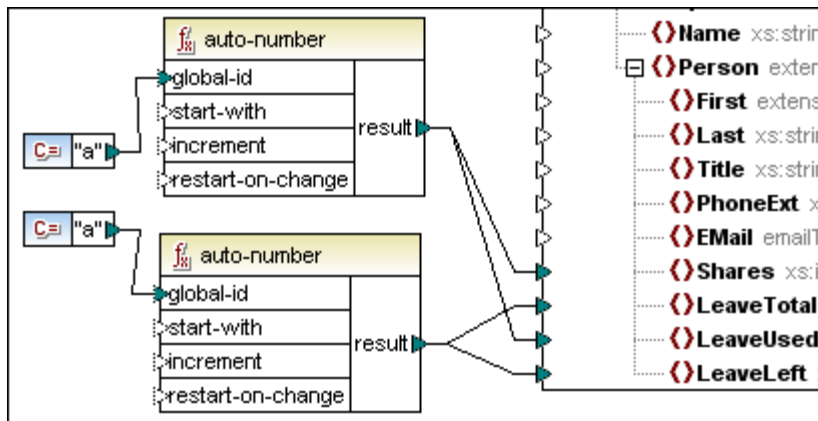
Wenn die beiden auto-number-Funktionen **nicht** dieselbe globale ID (global-id) haben, so inkrementiert jede davon die Ziel-Datenelemente separat. Im Beispiel unten hat jede Funktion eine andere global-id, nämlich a und b.

Die Ausgabe des Mappings ist 1,1,2,2. Die obere Funktion liefert die erste 1, die untere liefert die zweite 1.



Wenn beide Funktionen **identische** global-ids haben - in diesem Fall **a** - so "kennt" jede Funktion den aktuellen auto-number-Status (oder tatsächlichen Wert) der anderen und beide Zahlen werden anschließend der Reihe nach synchronisiert.

Die Ausgabe des Mappings ist daher 1, 2, 3, 4. Die obere Funktion liefert die erste 1 und die untere liefert nun eine 2.



### start-with

Der Anfangswert für den Beginn der automatisch nummerierten Sequenz. Der Standardwert ist 1.

### increment

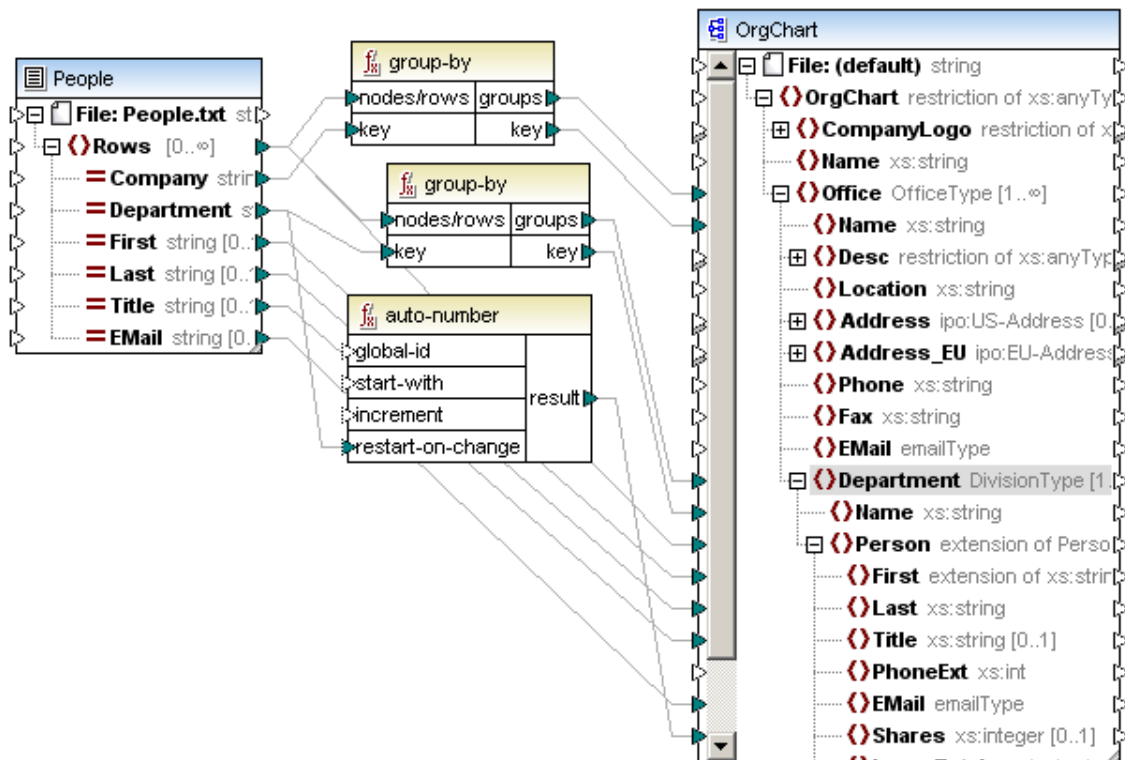
Der Inkrementierungswert, um den die automatisch nummerierte Sequenz erhöht werden soll. Der Standardwert ist 1.

### restart on change

Setzt den auto-number-Zähler auf "start-with" zurück, wenn sich der **Inhalt** des damit verbundenen Datenelements ändert.

Im Beispiel unten wird sowohl für start-with als auch für increment der Standardwert 1

verwendet. Sobald sich der **Inhalt** von "Department", also der Name der Abteilung, ändert, wird der Zähler zurückgesetzt und beginnt bei jeder neuen Abteilung mit 1.



### logical functions (logische Funktionen)

Logische Funktionen dienen (im Allgemeinen) dazu, zwei Gruppen von Input-Daten mit dem Ergebnis ("true" oder "false") zu vergleichen. Normalerweise werden Daten damit überprüft, bevor sie mittels eines Filters an eine Untergruppe der Zielkomponente übergeben werden.

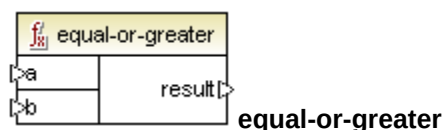
Beim Vergleich von Input-Parametern wählt MapForce den am engsten definierten gemeinsamen Typ für jeden Parameter aus und vergleicht diese dann. Wenn es sich beim gemeinsamen Typ um anySimpleType handelt, so werden beide Input-Parameter als Strings verglichen.

Input-Parameter = **a** | **b** oder **value1** | **value2**

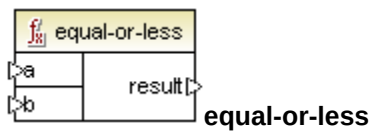
Output-Parameter = **result**



Das Ergebnis ist "true", wenn a=b, andernfalls "false".



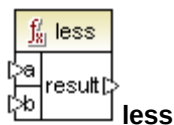
Das Ergebnis ist "true", wenn a gleich/größer als b ist, andernfalls ist es "false".



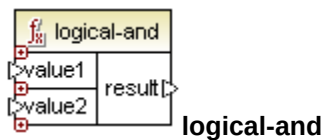
Das Ergebnis ist "true", wenn a gleich/kleiner als b ist, andernfalls ist es "false".



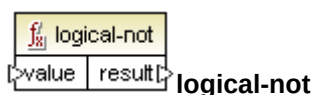
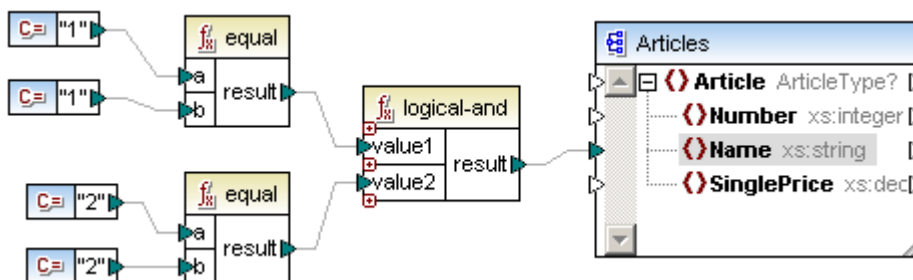
Das Ergebnis ist "true", wenn a größer als b ist, andernfalls ist es "false".



Das Ergebnis ist "true", wenn a kleiner als b ist, andernfalls ist es "false".

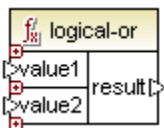
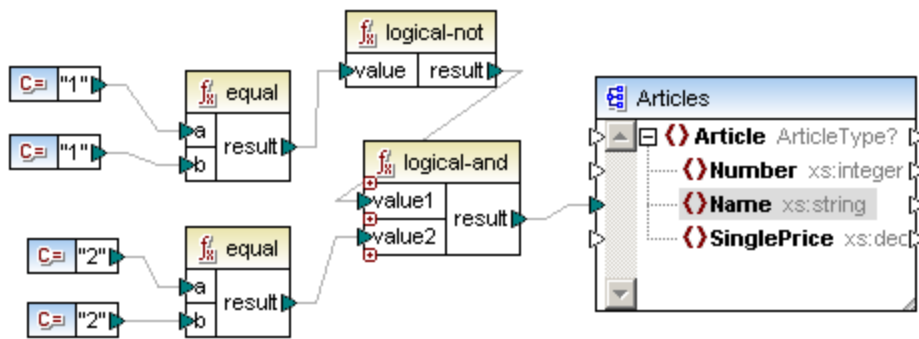


Wenn **sowohl** value1 **als auch** value2 der logical-and Funktion **true** ist, dann ist das Ergebnis true. Andernfalls ist es false.

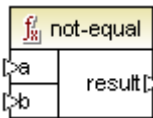


Invertiert oder spiegelt den logischen Zustand/das Ergebnis; wenn der Input-Wert "true" ist, ist das Ergebnis der logical-not-Funktion "false". Wenn der Input "false" ist, ist das Ergebnis "true".

In der unten gezeigten logical-not Funktion wird das Ergebnis der equal-Funktion umgekehrt. Die logical-and Funktion gibt nun nur dann "true" zurück, wenn die Booleschen Werte von value1 und value2 unterschiedlich sind, d.h. true-false oder false-true.

**logical-or**

Hierfür müssen beide Input-Werte Boolesche Werte sein. Wenn **entweder** value1 **oder** value2 der logical-or Funktion **true** ist, so ist das Ergebnis "true". Wenn beide Werte "false" sind, so ist das Ergebnis "false".

**not equal**

Das Ergebnis ist "true", wenn a nicht gleich b ist.

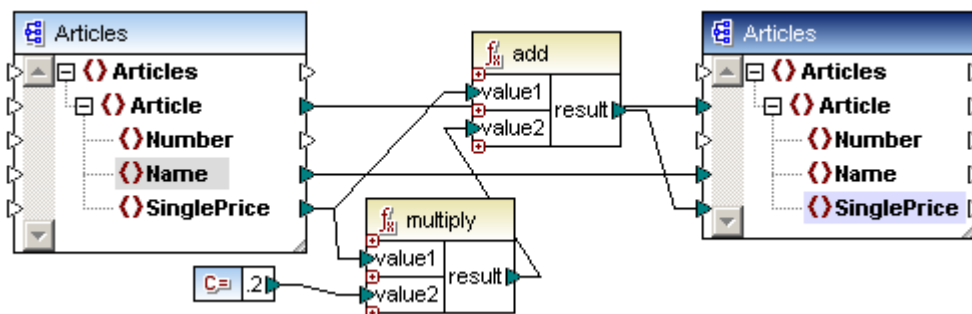
**math functions (mathematische Funktionen)**

Mathematische Funktionen dienen zur Durchführung grundlegender mathematischer Funktionen an Daten. Beachten Sie, dass diese Funktionen nicht zur Berechnung einer Zeitdauer oder zur Berechnung von Datum und Uhrzeit verwendet werden können.

Input-Parameters = **value1** | **value2**

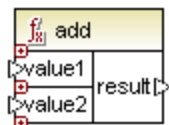
Output-Parameter = **result**

Input-Werte werden zur weiteren Verarbeitung automatisch in Dezimalwerte (decimal) konvertiert.

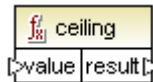


Im obigen Beispiel werden 20 % Ust zu den einzelnen auf die Zielkomponente gemappten Artikeln hinzugefügt.



**add**

Das Ergebnis ist der Dezimalwert der Addition von value1 zu value2.

**ceiling**

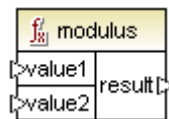
Das Ergebnis ist die kleinste Ganzzahl, die größer oder gleich dem nächsthöheren Ganzzahlwert des Input-Dezimalwert ist. Wenn z.B. das Ergebnis einer Divisionsfunktion 11,2 ist, ist das Ergebnis durch Anwendung der ceiling-Funktion 12, also die nächsthöhere Ganzzahl.

**divide**

Das Ergebnis ist der Dezimalwert der Division von value1 durch value2. Die Präzision des Ergebnisses hängt von der Zielsprache ab. Mit Hilfe der [round-precision](#) Funktion können Sie die Präzision des Ergebnisses definieren.

**floor**

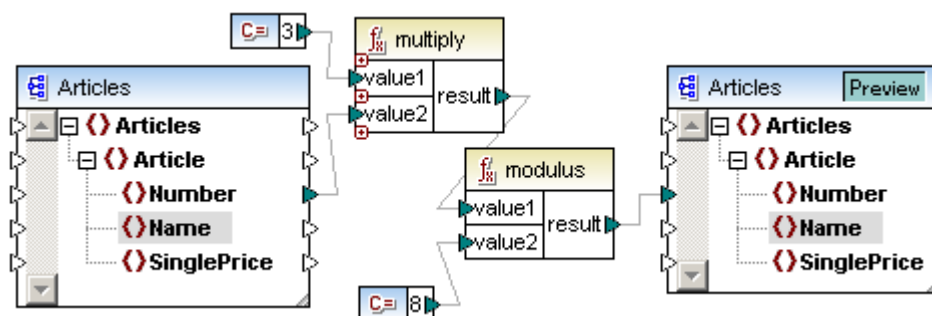
Das Ergebnis ist der nächstniedrigere Ganzzahlwert, der kleiner oder gleich **value** ist, d.h. der nächstniedrigere Ganzzahlwert des Input-Dezimalwerts. Wenn also das Ergebnis der Divisionsfunktion 11,2 ist, so wird das Ergebnis durch Anwendung der floor-Funktion zu 11, d.h. der nächstniedrigeren Ganzzahl.

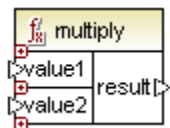
**modulus**

Das Ergebnis ist der Rest, der sich bei der Division von **value1** durch **value2** ergibt.

Im unten gezeigten Mapping wurden die Zahlen (Number) mit 3 multipliziert und an value1 der modulus-Funktion übergeben. Die Input-Werte sind nun 3, 6, 9 und 12.

Bei Anwendung/Verwendung von modulus 8 als value2, sind die Restwerte 3, 6, 1 und 4.

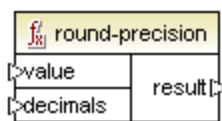


**multiply**

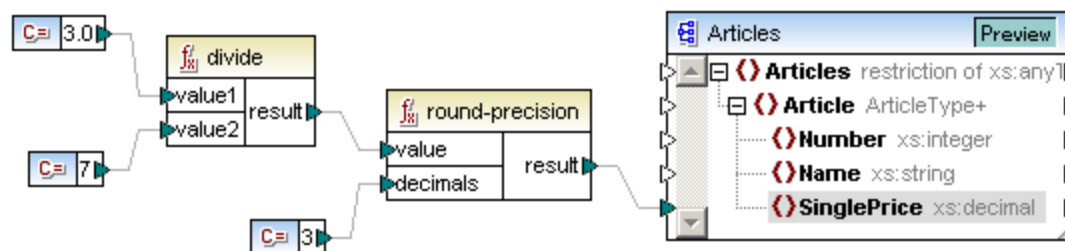
Das Ergebnis ist der Dezimalwert der Multiplikation von value1 mit value2.

**round**

Das Ergebnis ist der Rundungswert auf die nächste Ganzzahl.

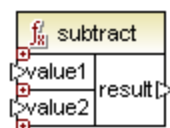
**round-precision**

Das Ergebnis ist der Dezimalwert der Zahl, gerundet auf die durch "decimals" definierten Dezimalstellen.



Im oben gezeigten Mapping ist das Ergebnis = 0,429.

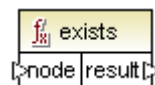
Wenn dieses Ergebnis in einer XML-Datei korrekt angezeigt werden soll, muss es auf ein Element vom Typ xs:decimal gemappt werden.

**subtract**

Das Ergebnis ist der Dezimalwert der Subtraktion von value2 von value1.

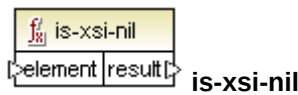
**node functions (Node-Funktionen)**

Mit Hilfe der Node Test-Funktionen können Sie überprüfen, ob bestimmte Nodes in vielen verschiedenen Arten von Input-Dateien, XML-Schema, Textdateien, Datenbanken, EDI-Dateien und sogar Funktionsergebnissen vorhanden sind. exists überprüft auf nicht leere Sequenzen, d.h. ob ein Node vorhanden ist.

**exists**

Gibt "true" zurück, wenn der Node vorhanden ist und andernfalls "false".

Ein Beispiel dazu finden Sie unter [exists](#).

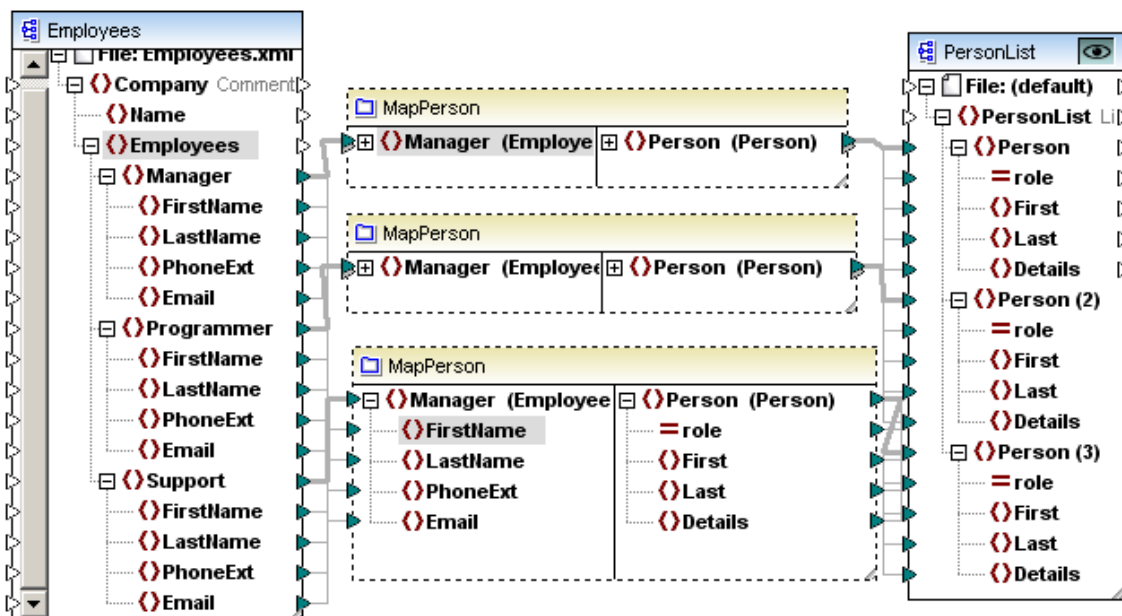


Gibt true (`<OrderID>true</OrderID>`) zurück, wenn das xsi:nil-Attribut für den Element-Node der Quellkomponente auf "true" gesetzt ist.

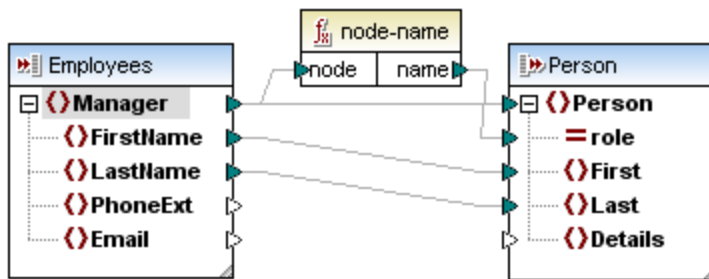


Gibt den QName des verbundenen Node zurück, außer es handelt sich um einen XML text()-Node. Wenn dies der Fall ist, wird ein leerer QName zurückgegeben. Die Funktion funktioniert nur mit Nodes, die einen Namen haben. Wenn XSLT die Zielsprache ist (die fn:node-name aufruft), wird eine leere Sequenz für Nodes, die keinen Namen haben, zurückgegeben.

- Das Abrufen eines Namens aus Datenbanktabellen bzw. -feldern wird nicht unterstützt.
- XBRL und Excel werden nicht unterstützt
- Das Abrufen eines Names des Datei-Input-Node wird nicht unterstützt
- Webservice-Nodes werden wie XML-Nodes behandelt mit folgenden Ausnahmen:
  - node-Name anhand eines Teils wird nicht unterstützt.
  - node-Name anhand des Root Node ("Output" oder "Input") wird nicht unterstützt



In der benutzerdefinierten Funktion "MapPerson" wird **node-name** verwendet, um den Namen des Input-Node zurückzugeben und in das Role-Attribut einzusetzen. Als Root-Node der Datei Employees.xsd wurde in der benutzerdefinierten Input-Funktion "Manager" definiert.



Manager erhält seine Daten wie oben gezeigt von **außerhalb** der benutzerdefinierten Funktion, wo es entweder Manager, Programmierer oder Support sein kann. Dies sind die Daten, die anschließend an das role-Attribut in PersonList übergeben werden.



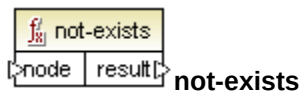
Gibt den String mit dem Namen des damit verbundenen Node zurück. Beim Input muss es sich: (i) um eine [Quellkomponente](#) oder (ii) um eine [Inline-Funktion](#) handeln, die direkt mit einem [Parameter](#) verbunden ist, der wiederum direkt mit einem Node im aufrufenden Mapping verbunden ist.

Die Verbindung muss direkt erfolgen und darf nicht über einen Filter oder eine benutzerdefinierte Nicht-Inline-Funktion laufen. Dies ist eine Pseudofunktion, die bei der Generierung durch den aus dem verbundenen Node abgerufenen Text ersetzt wird und daher für alle Programmiersprachen zur Verfügung steht.



Gibt den String mit der Annotation des verbundenen Node zurück. Beim Input muss es sich: (i) um eine [Quellkomponente](#) oder (ii) um eine [Inline-Funktion](#) handeln, die direkt mit einem [Parameter](#) verbunden ist, der wiederum direkt mit einem Node im aufrufenden Mapping verbunden ist.

Die Verbindung muss direkt erfolgen und darf nicht über einen Filter oder eine benutzerdefinierte Nicht-Inline-Funktion laufen. Dies ist eine Pseudofunktion, die bei der Generierung durch den aus dem verbundenen Node abgerufenen Text ersetzt wird und daher für alle Programmiersprachen zur Verfügung steht.



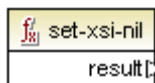
Gibt "false" zurück, wenn der Node vorhanden ist und andernfalls "true".

Ein Beispiel dazu finden Sie unter [not-exists](#).

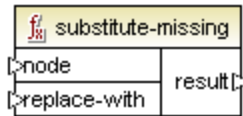


Gibt die Position eines bestimmten Node in einer Sequenz zurück.

Ein Beispiel dazu finden Sie unter [position](#).

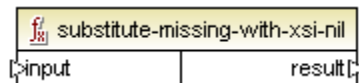
**set-xsi-nil**

Setzt den Ziel-Node auf xsi:nil.

**substitute-missing**

Diese Funktion ist eine praktische Kombination aus "exists" und einer passenden "if-else" Bedingung. Dient zum Mappen des aktuellen Feldinhalts, wenn der Node in der XML-Quelldatei existiert. Andernfalls wird das Datenelement auf den Parameter "replace-with" gemappt.

Ein Beispiel dazu finden Sie unter [substitute-missing](#).

**substitute-missing-with-xsi-nil**

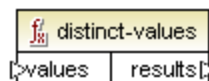
Ersetzt alle fehlenden (oder Null-Werte) der Quellkomponente im Ziel-Node durch das xsi:nil Attribut.

**sequence functions (Sequenzfunktionen)**

Sequenzfunktionen stehen nicht zur Verfügung, wenn XSLT (XSLT 1.0) ausgewählt ist.

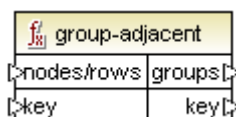
MapForce unterstützt nun Sequenzfunktionen. Mit Hilfe von Sequenzfunktionen können Input-Sequenzen verarbeitet und ihr Inhalt gruppiert werden. Um die Funktion zu gruppieren, wird Wert/Inhalt des Input-Parameters **key** auf Nodes/Zeilen gemappt.

- der Input-Parameter **key** hat bei **group-adjacent** und **group-by** einen beliebigen Datentyp, der in "string" konvertiert werden kann.
- der Input-Parameter **bool** hat bei **group-starting-with** und **group-ending-with** den Typ Boolean.
- Der Output **key** ist der Schlüssel der aktuellen Gruppe.

**distinct-values**

Dient zum Entfernen doppelt vorhandener Werte aus einer Sequenz und zum Mappen der eindeutigen Datenelemente auf die Zielkomponente.

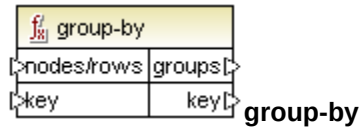
Ein Beispiel dazu finden Sie unter [distinct-values](#).

**group-adjacent**

Gruppiert die Input-Sequenz **nodes/rows** in Gruppen benachbarter Datenelemente, die denselben **key** (Schlüssel) haben.

Beachten Sie, dass group-adjacent den **Inhalt** des Node/Datenelements als Gruppierungsschlüssel verwendet!

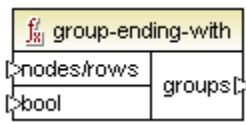
Ein Beispiel dazu finden Sie unter [group-adjacent](#).



### group-by

Gruppiert die Input-Sequenz **nodes/rows** in Gruppen nicht unbedingt benachbarter Datenelemente, die denselben **key** (Schlüssel) haben. Die Gruppen werden in der Reihenfolge ausgegeben, in der der Schlüssel (key) in der Input-Sequenz vorkommt.

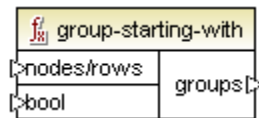
Ein Beispiel dazu finden Sie unter [group-by](#).



### group-ending-with

Diese Funktion Gruppiert die Input-Sequenz **nodes/rows** in Gruppen und beendet jede neue Gruppe, wenn **bool** true ist.

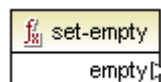
Ein Beispiel dazu finden Sie unter [group-ending-with](#).



### group-starting-with

Diese Funktion Gruppiert die Input-Sequenz **nodes/rows** in Gruppen und beginnt jede neue Gruppe, wenn **bool** true ist.

Ein Beispiel dazu finden Sie unter [group-starting-with](#).

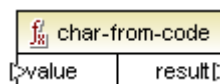


### set-empty

Dient zum Löschen von [Standardwerten](#) eines XBRL-Dokuments, die weiter oben in der XBRL-Komponente/Taxonomie definiert wurden.

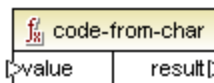
## string functions (String-Funktionen)

Mit diesen Funktionen können Sie viele Quelldatenarten mit Hilfe der am häufigsten verwendeten String-Funktionen bearbeiten um Abschnitte zu extrahieren, den String auf darin enthaltene Strings zu überprüfen oder Informationen über Strings abzurufen.

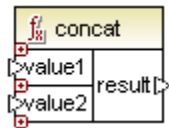


### char-from-code

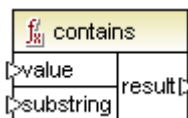
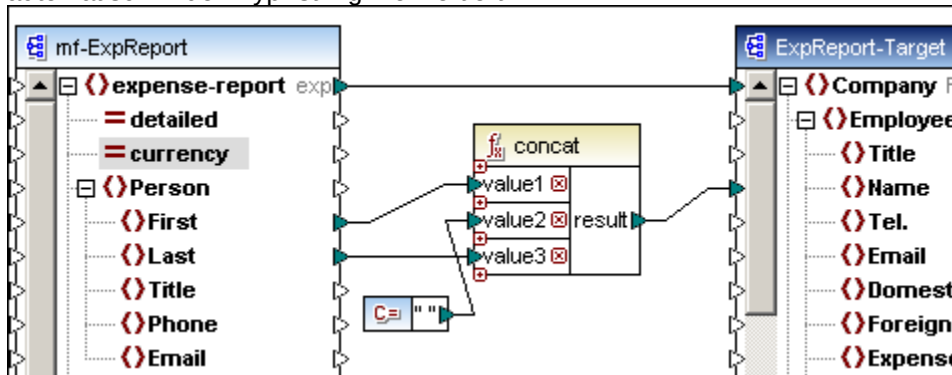
Das Ergebnis die Zeichendarstellung des Unicode-Dezimalwerts von **value**.

**code-from-char**

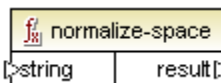
Das Ergebnis ist der Unicode-Dezimalwert des ersten Zeichens von **value**.

**concat**

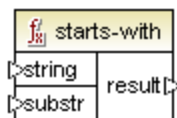
Verkettet zwei oder mehr Werte zu einem einzigen Ergebnisstring. Alle Input-Werte werden automatisch in den Typ "string" konvertiert.

**contains**

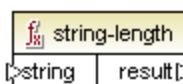
Das Ergebnis ist "true", wenn die an den Parameter "value" gelieferten Daten den im Substring-Parameter definierten String enthalten.

**normalize-space**

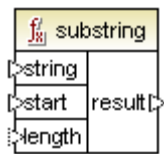
Das Ergebnis ist der normalisierte Input-String, d.h. vorangestellte und nachgestellte Leerzeichen werden entfernt, mehrere Whitespace-Zeichen werden zu einem einzigen Whitespace-Zeichen zusammengefasst.

**starts-with**

Das Ergebnis ist "true", wenn der Input-String "string" mit substr beginnt, andernfalls ist es "false".

**string-length**

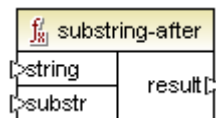
Das Ergebnis ist die Anzahl der vom String-Parameter gelieferten Zeichen.

**substring**

Das Ergebnis ist der Substring (das String-Fragment) des Parameters "string", wobei "start" die Position des Anfangszeichens und "length" die Länge des Substring definiert.

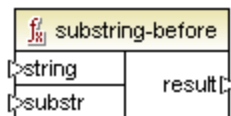
Wenn der Parameter "length" nicht definiert ist, ist das Ergebnis ein Fragment, das an der Anfangsposition des String beginnt und an der Endposition des String endet.

z.B. substring("56789",2,3) hat das Ergebnis 678.

**substring-after**

Das Ergebnis ist der Rest des Parameters "**string**", wobei die Stelle, an der der Parameter "**substr**" das erste Mal vorkommt, das Anfangszeichen definiert; der Rest des String ist das Ergebnis der Funktion. Das Ergebnis ist ein leerer String, wenn **substr** in **string** nicht vorkommt.

z.B. hat substring-after("2009/01/04","/") als Ergebnis den Substring 01/04. substr ist in diesem Fall das erste "/" Zeichen.

**substring-before**

Das Ergebnis ist das String-Fragment des Parameters "string" bis zu dem ersten Zeichen, an dem die Zeichen des Substring zum ersten Mal vorkommen. Das Ergebnis ist ein leerer String, wenn **substr** in **string** nicht vorkommt.

z.B. substring-before ("2009/01/04","/") hat als Ergebnis den Substring 2009. substr ist in diesem Fall das erste "/" Zeichen.

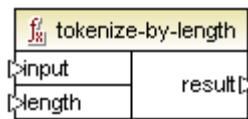
**tokenize**

Als Ergebnis wird der Input-String in eine Sequenz von Blöcken/Abschnitten aufgeteilt, die durch den Parameter "delimiter" definiert werden. Anschließend kann das Ergebnis zur Weiterverarbeitung weitergegeben werden.

Beispiel: Der Input-String ist A,B,C und der "delimiter" (das Trennzeichen) ist "," - das Ergebnis ist A B C.

Ein Beispiel dazu finden Sie unter [Tokenize Beispiele](#).

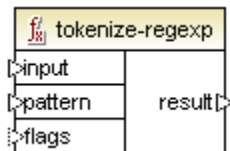


**tokenize-by-length**

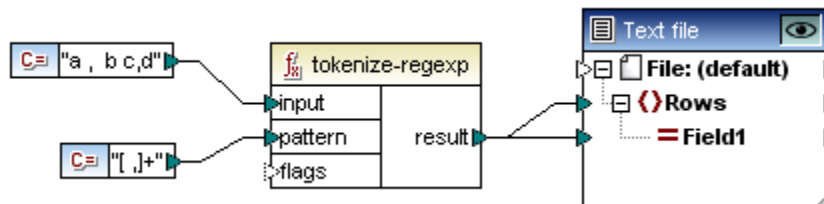
Als Ergebnis wird der Input-String in eine Anzahl von Blöcken/Abschnitten aufgeteilt, die durch den Parameter "length" definiert werden. Anschließend kann das Ergebnis zur Weiterverarbeitung weitergegeben werden.

Beispiel: der Input-String ist ABCDEF und length ist "2" - dann ist das Ergebnis AB CD EF.

Ein Beispiel dazu finden Sie unter [Tokenize Beispiele](#).

**tokenize-regexp**

Als Ergebnis der Funktion wird der Input-String in eine Reihe von Strings aufgeteilt, wobei das Ergebnis des definierten Regular Expression Pattern (Muster) als Trennzeichen fungiert. Die Trennzeichenstrings werden nicht vom result-Parameter ausgegeben. Es können auch optionale Flags verwendet werden.



Im oben gezeigten Beispiel:

ist der **input** String ist eine Folge von Zeichen, die durch Leerzeichen und/oder Kommas getrennt sind, also a , b c,d

definiert das regex **pattern** definiert eine Zeichenklasse ["space""comma"], von der nur ein einziges Zeichen in einer Zeichenklasse verwendet wird, also entweder space (Leerzeichen) oder comma (Komma).

definiert der + Quantifizierer "eine oder mehrere" Instanzen der Zeichenklasse/des Strings.

Der Ergebnisstring ist:

|   |   |
|---|---|
| 1 | a |
| 2 | b |
| 3 | c |
| 4 | d |
| 5 |   |

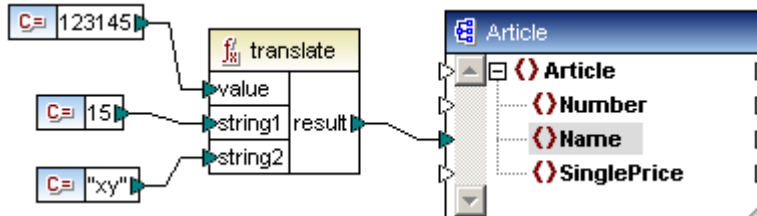
Bitte beachten Sie: Die Syntax von Regular Expressions unterscheidet sich in den verschiedenen Sprachen geringfügig voneinander. Die Funktion Tokenize-regexp steht in C++ nur in Visual Studio 2008 und höheren Versionen zur Verfügung.

Nähere Informationen zu Regular Expressions finden Sie unter: [Regular Expressions](#).

**translate**

Die Zeichen von **string1** (Suchstring) werden im Input-String **value** durch die an derselben Position stehenden Zeichen in **string2**(Ersetzungsstring) ersetzt.

Wenn es in String2 keine entsprechenden Zeichen gibt, wird das Zeichen entfernt.

**Beispiel**

der Input-String ist 123145  
 der (Such)string1 ist 15  
 (der Ersetzungs)string2 ist xy

Folglich:

wird jede 1 im Input-String-Wert durch **x** ersetzt  
 und jede 5 wird im Input-String durch **y** ersetzt

Der Ergebnisstring ist **x23x4y**

Wenn string2 leer ist (weniger Zeichen als string1), so wird das Zeichen entfernt.

**Beispiel B. 2**

Der Input-String aabaacbca  
 string1 ist "a"  
 string2 ist "" (leerer String)

Der Ergebnisstring ist "bcbc"

**Beispiel 3**

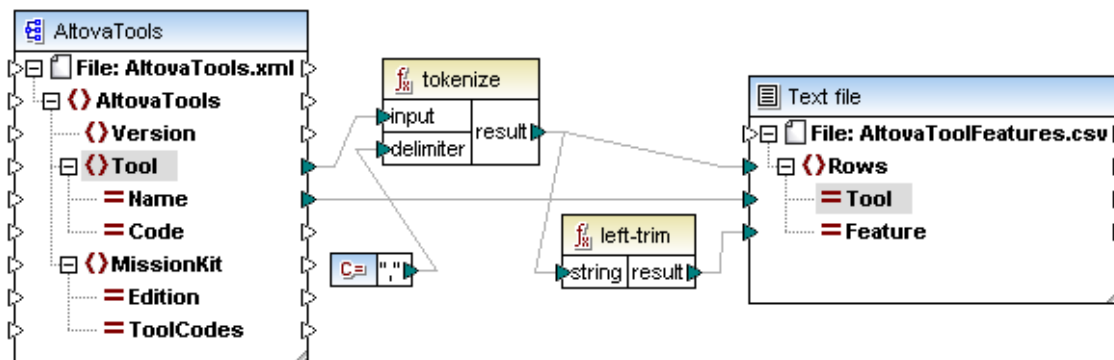
Der Input-String aabaacbca  
 string1 ist "ac"  
 string2 ist "ca"

Der Ergebnisstring lautet "cbccabac"

Tokenize Beispiele

**Beispiel tokenize:**

In der Datei **tokenizeString1.mfd** im Ordner ...\\MapForceExamples sehen Sie, wie die tokenize-Funktion verwendet wird.



Unten sehen Sie die XML-Quelldatei. Das Element **Tool** hat zwei Attribute: Name und Code, wobei die Daten des Elements "Tool" aus durch Kommas getrenntem Text bestehen.

AltovaTools

xmlns:xsi

http://www.w3.org/2001/XMLSchema-instance

xsi:noName...

AltovaTools.xsd

Version

2010

Tool (9)

#### Funktionsweise des Mappings:

- Die tokenize-Funktion erhält Daten aus dem Datenelement **Tool** und teilt die Daten mit Hilfe des Komma-Trennzeichens ";" in separate Blöcke auf. D.h., der erste Block ist "XML editor".
- Der result-Parameter wird auf das Datenelement **Rows** in der Zielkomponente gemappt. Dies wird für alle Elemente in der Quelldatei durchgeführt.
- Der result-Parameter wird außerdem auch auf die **left-trim** Funktion gemappt, die aus den einzelnen Blöcken die vorangestellten Leerzeichen entfernt.
- Das Ergebnis des left-trim-Parameters (die einzelnen Blöcke) werden auf das Datenelement **Feature** der Zielkomponente gemappt.
- Als Ausgabedatei der Zielkomponente wurde eine CSV-Datei definiert (AltovaToolFeatures.csv), in der als Feldtrennzeichen das Semikolon verwendet wird (Doppelklicken Sie auf die Komponente, um die Einstellungen zu sehen).

#### Ergebnis des Mappings:

- Für jedes Tool-Element der Quelldatei
- wird der (Tool) Name auf das Datenelement "Tool" in der Zielkomponente gemappt
- werden die einzelnen Blöcke des aufgeteilten Inhalts von "Tool" an das (Tool Name) Feature-Datenelement angehängt.
- So erhält z.B. des erste Tool, XMLSpy, den ersten Feature-Block "XML editor"
- Dieser Vorgang wird für alle Blöcke des aktuellen Tools wiederholt und anschließend

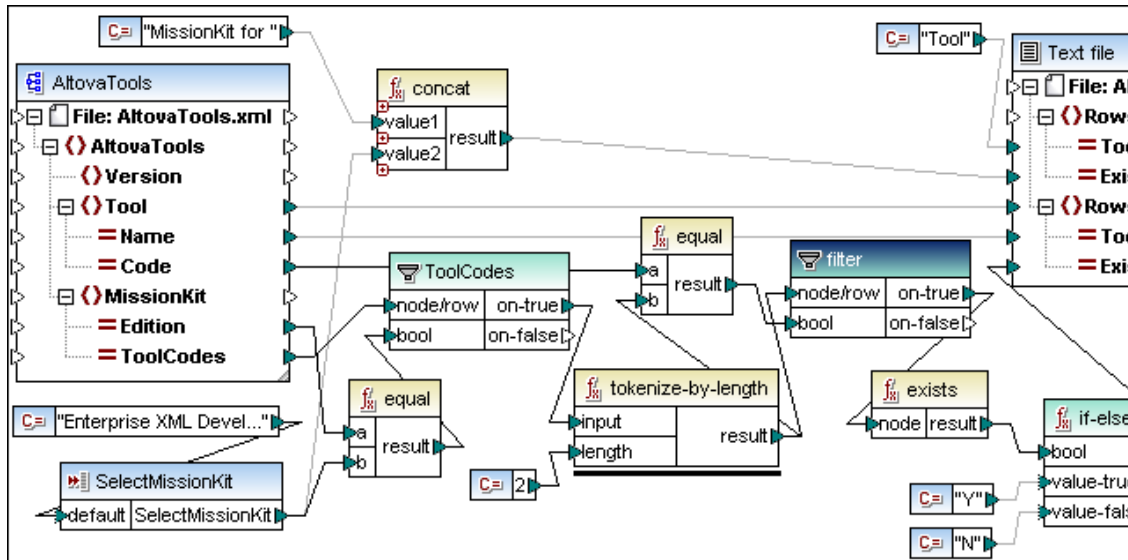
für alle Tools.

- Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie das unten gezeigte Ergebnis.

|    |                                |
|----|--------------------------------|
| 1  | Tool;Feature                   |
| 2  | XMLSpy;XML editor              |
| 3  | XMLSpy;XSLT editor             |
| 4  | XMLSpy;XSLT debugger           |
| 5  | XMLSpy;XQuery editor           |
| 6  | XMLSpy;XQuery debugger         |
| 7  | XMLSpy;XML Schema / DTD editor |
| 8  | XMLSpy;WSDL editor             |
| 9  | XMLSpy;SOAP debugger           |
| 10 | MapForce;Data integration      |
| 11 | MapForce;XML mapping           |
| 12 | MapForce;database mapping      |

## Beispiel tokenize-by-length

In der Datei **tokenizeString2.mfd** im Ordner ...MapForceExamples sehen Sie, wie die **tokenize-by-length** Funktion verwendet wird.



Die unten gezeigte XML-Quelldatei ist dieselbe wie die im vorhergehenden Beispiel verwendete. Das Element **MissionKit** hat ebenfalls zwei Attribute: **Edition** und **ToolCodes** aber keinen Inhalt für das Element "MissionKit".

| Tool (9) |               |      |                                                                                   |
|----------|---------------|------|-----------------------------------------------------------------------------------|
|          | Name          | Code | Abc Text                                                                          |
| 1        | XMLSpy        | XS   | XML editor, XSLT editor, XSLT debugger, XQuery editor, XQuery debugger, XML S     |
| 2        | MapForce      | MF   | Data integration, XML mapping, database mapping, text conversion, EDI translator, |
| 3        | StyleVision   | SV   | Stylesheet designer, electronic forms, XSLT design, XSL:FO design, database rep   |
| 4        | UModel        | UM   | UML modeling tool, code generation, reverse engineering, UML, BPMN, SysML, pro    |
| 5        | DatabaseSpy   | DS   | Multi-database tool, SQL auto-completion, graphical database design, table brows  |
| 6        | DiffDog       | DD   | Diff / merge tool, compare files, sync directories, compare XML, compare OOXML    |
| 7        | SchemaAgent   | SA   | XML Schema management tool, IIR management, XSLT management, WSDL manag           |
| 8        | SemanticWorks | SV   | Semantic Web tool, RDF editor, OWL editor, RDF/XML and N-Triples generation and   |
| 9        | Authentic     | AU   | XML authoring tool, database editor, XML publishing tool, e-Forms editor          |

| MissionKit (4) |                                  |                 |
|----------------|----------------------------------|-----------------|
|                | Edition                          | ToolCodes       |
| 1              | Enterprise Software Architects   | XSMFSVUMDSDDASW |
| 2              | Professional Software Architects | XSMFSVUMDS      |
| 3              | Enterprise XML Developers        | XSMFSVDDASW     |
| 4              | Professional XML Developers      | XSMFSV          |

## Ziel des Mappings:

Ziel ist die Generierung einer Liste, in der Sie sehen, welche Altova-Tools jeweils Teil der entsprechenden MissionKit Editions sind.

## Funktionsweise des Mappings:

- Die SelectMissionKit **input**-Komponente erhält ihren Standard-Input von einer Konstantenkomponente, in diesem Fall von "Enterprise XML Developers".
- Mit Hilfe der **equal**-Funktion wird der Input-Wert mit dem Wert von "**Edition**" verglichen. Das Ergebnis wird an den **bool**-Parameter des ToolCodes-Filters übergeben.

- Der **node/row**-Input des ToolCodes-Filters stammt aus dem Quelldatei-Datenelement **ToolCodes**. Der Wert für die Enterprise XML Developers Edition ist: XSMFSVDDASW.
- Der Wert XSMFSVDDASW wird an den **on-true**-Parameter übergeben und dann an den **input**-Parameter der **tokenize-by-length** Funktion.

#### Funktionsweise und Ergebnis der **tokenize-by-length** Funktion:

- Der ToolCodes **input**-Wert XSMFSVDDASW wird in mehrere Blöcke bestehend aus je zwei Zeichen aufgeteilt, die durch den **length** Parameter 2 definiert werden. Das Ergebnis sind 6 Blöcke.
- Jeder (in den Parameter b platzierte) Block der equal-Funktion wird mit dem aus 2 Zeichen bestehenden **Code**-Wert der Quelldatei verglichen (von der es insgesamt 9 Einträge/Datenelemente gibt).
- Das Ergebnis des Vergleichs (true/false) wird an den **bool**-Parameter des Filters übergeben.
- Beachten Sie, dass **alle** Blöcke der **tokenize-by-length** Funktion an den **node/row**-Parameter des Filters übergeben werden.

- Die **exist**-Funktion überprüft nun die Datei auf existing/non-existing (existierende/nicht existente) Nodes, die vom **on-true** Parameter der Filter-Komponente an die **exist**-Funktion übergeben werden.

Existierende Nodes sind jene, für die eine **Übereinstimmung** zwischen dem **ToolCodes**-Block und dem Codewert besteht.

Nicht existente Nodes sind Fälle, in denen es **keine Übereinstimmung** zwischen einem **ToolCodes**-Block und einem Codewert gibt.

- Die bool-Ergebnisse der **exists**-Funktion werden an die **if-else** Funktion übergeben, die ein Y an das Ziel übergibt, wenn der Node vorhanden ist oder ein N, wenn der Node nicht existiert.

Ergebnis des Mappings:

|    |                                               |
|----|-----------------------------------------------|
| 1  | Tool;MissionKit for Enterprise XML Developers |
| 2  | XMLSpy;Y                                      |
| 3  | MapForce;Y                                    |
| 4  | StyleVision;Y                                 |
| 5  | UModel;N                                      |
| 6  | DatabaseSpy;N                                 |
| 7  | DiffDog;Y                                     |
| 8  | SchemaAgent;Y                                 |
| 9  | SemanticWorks;Y                               |
| 10 | Authentic;N                                   |
| 11 |                                               |

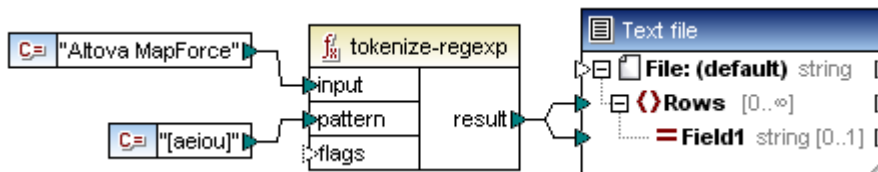
#### Regular Expressions

MapForce kann Regular Expressions im **pattern**-Parameter der Funktionen [match-pattern](#) und [tokenize-regexp](#) verwenden, um bestimmte Strings des Input-Parameters zu finden.

Die Syntax und Semantik von Regular Expressions für XSLT und XQuery entspricht den in <http://www.w3.org/TR/xmlschema-2/> definierten Regeln. Beachten Sie bitte, dass es in der Syntax von Regular Expressions geringfügige Unterschiede zwischen den verschiedenen Programmiersprachen gibt.

**Terminologie:**

|         |                                                                                          |
|---------|------------------------------------------------------------------------------------------|
| input   | der String, an dem der regex-Ausdruck ausgeführt wird                                    |
| pattern | die Regular Expression                                                                   |
| flags   | optionaler Parameter, um zu definieren, wie die Regular Expression zu interpretieren ist |
| result  | das Ergebnis der Funktion                                                                |



Tokenize-regex gibt eine Stringsequenz zurück. Mit der Verbindung des Datenelements "Rows" wird in der Sequenz eine Zeile pro Datenelement erstellt.

**regex-Syntax**

**Literals** e.g. Einzelnes Zeichen:

"a" ist z.B. der einfachste regex-Ausdruck. Er findet die erste Instanz des Buchstaben "a" in einem String.

**Zeichenklassen []**

Dies ist eine Gruppe von Zeichen, die innerhalb von eckigen Klammern steht.

**Es wird immer nur jeweils ein einziges** der Zeichen innerhalb der eckigen Klammern gefunden.

pattern **[aeiou]**

Findet einen kleingeschriebenen Vokal.

pattern **[mj]ust**

Findet must oder just.

Bitte beachten Sie, dass bei "pattern" die Groß- und Kleinschreibung eine Rolle spielt, daher wird mit einem kleingeschriebenen **a** der Großbuchstabe **A** nicht gefunden.

**Zeichenbereich [a-z]**

Erstellt einen Bereich zwischen den beiden Zeichen. Es wird immer nur jeweils eines der Zeichen gefunden.

pattern **[a-z]**

Findet alle Kleinbuchstaben zwischen a und z.

**Negierte Klassen [^]**

mit dem Zirkumflexzeichen als erstem Zeichen nach der öffnenden Klammer wird die Zeichenklasse negiert.

pattern **[^a-z]**

Findet alle Zeichen außerhalb der Zeichenklasse, einschließlich von Zeilenschaltungen

(newline).

### Metazeichen "."

Punkt-Metazeichen

steht für **jedes beliebige einzelne** Zeichen (mit Ausnahme von "newline")

pattern .

Findet jedes einzelne Zeichen.

### Quantifizierer ? + \* {}

Quantifizierer definieren, wie oft eine regex-Komponente im Input-String wiederholt werden muss, damit eine Übereinstimmung gefunden wird.

**?**

null oder eins      der vorhergehende String/Abschnitt ist optional

**+**

eines oder mehrere der vorhergehende String/Abschnitt darf ein Mal oder öfter gefunden werden

**\***

null oder mehr      der vorhergehende String/Abschnitt darf null Mal oder öfter gefunden werden

**{}**

min / max      die Anzahl der Wiederholungen mit denen ein String/  
Wiederholungen      Abschnitt gefunden werden muss  
z.B. wird mit "mo{1,3}" mo, moo, mooo gefunden.

**()**

Subpatterns (Untergruppen)

Um Teile eines regex-Ausdrucks zu gruppieren, werden Klammern verwendet.

**|**

Alternation/oder      gestattet das Testen von Unterausdrücken von links nach rechts.  
**(horse|make) sense** - findet "horse sense" oder "make sense"

### Flags

Flags sind optionale Parameter, die definieren, wie die Regular Expression interpretiert werden soll. Zum Definieren der Optionen dienen einzelne Buchstaben, d.h., wenn das jeweilige Zeichen vorhanden ist, ist diese Option aktiv. Die Buchstaben können in jeder Reihenfolge vorkommen und auch wiederholt werden.

**s**

Falls vorhanden, wird der Suchvorgang im "dot-all"-Modus durchgeführt.

Das Metazeichen "." steht für jedes beliebige Zeichen. Wenn der String "hello" und "world" in zwei *verschiedenen* Zeilen vorkommt, wird mit der Regular Expression "hello\*world" nur gefunden, wenn das Flag/Zeichen **s** gesetzt wurde.

**m**

Falls vorhanden, wird der Suchvorgang im mehrzeiligen Modus durchgeführt.



Im mehrzeiligen Modus findet das Zirkumflexzeichen ^ den Beginn **jeder** Zeile, d.h. den Beginn des gesamten Strings **und** das erste Zeichen nach dem "newline" Zeichen.

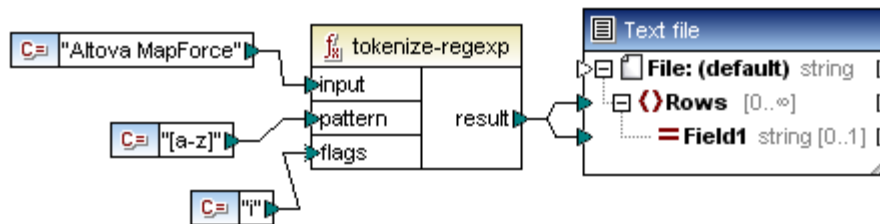
Das Dollarzeichen \$ findet das Ende **jeder** Zeile, d.h. das Ende des gesamten Strings und das Zeichen unmittelbar vor dem "newline" Zeichen.

Newline (Neue Zeile) ist das Zeichen #x0A.

### **i**

Falls vorhanden, wird der Suchvorgang im Modus "Groß-/Kleinschreibung ignorieren" durchgeführt.

Mit der Regular Expression [a-z] plus dem i Flag würden alle Buchstaben von a-z und A-Z gefunden.



### **x**

Falls vorhanden, werden Leerzeichen und andere Whitespace-Zeichen vor dem Suchvorgang aus der Regular Expression entfernt. Die Whitespace-Zeichen sind #x09, #x0A, #x0D und #x20.

*Ausnahme:*

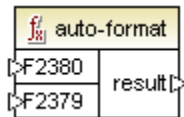
Whitespace-Zeichen innerhalb von Zeichenklassen-Ausdrücken werden nicht entfernt z.B. [#x20].

Bitte beachten Sie:

Die komplexen Funktionalitäten der regex-Syntax können sich bei der Generierung von Code in den verschiedenen Programmiersprachen etwas unterschiedlich verhalten. Lesen Sie bitte in der entsprechenden regex-Dokumentation zu Ihrer Sprache nach.

### 23.5.2 edifact

Die edifact-Bibliothek enthält Funktionen, die EDI-Datums-, Uhrzeit- und Zeitraumwerte in xs:date Formate konvertieren.



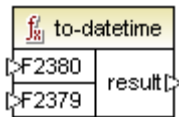
**auto-format**

Das Ergebnis ist der aus der kodierten Quelle (F2380) extrahierte Datums- / Uhrzeit- / Datum- und-Uhrzeit-Wert, der den Formatcode (F2379) erhält.



**to-date**

Das Ergebnis ist der aus der kodierten Quelle (F2380) extrahierte Datumswert, der den Formatcode (F2379) erhält.



**to-datetime**

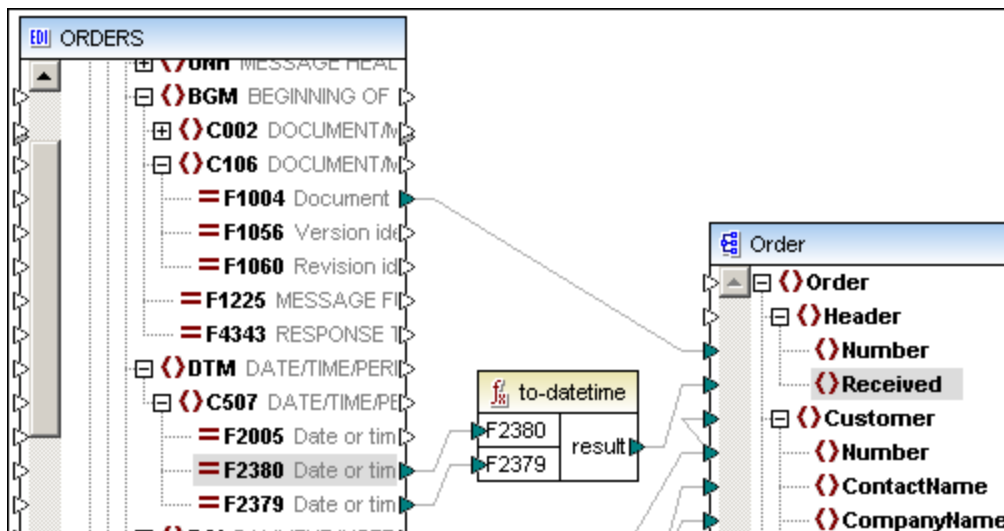
Das Ergebnis ist der aus der kodierten Quelle (F2380) extrahierte Datums- und Uhrzeitwert, der den Formatcode (F2379) erhält.

Beispiel:

In der EDI-Datei in der Abbildung unten ist der Datums-/Uhrzeit-/Datums- und Uhrzeitwert im Feld "DTM" enthalten.

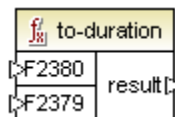
```
UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
ADDR+970101:1230+000000000000001++ORDERS++++1'
UNH+0001+ORDERS:D:08A:UN'
BGM+221+ABC123456XYZ+9'
DTM+4:200404301742PDT:303'
FTX+PUR+3++Pizza purchase order'
RFF+CT:123-456'
RFF+CR:1122'
NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+Ca
CTA+SR+:Ted Little'
```

Der Wert wird mit Hilfe einer to.datetime-Funktion in ein xs:date Format konvertiert.



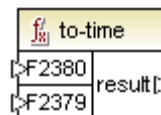
Das Ergebnis der Konvertierung sehen Sie unten. Dieses Beispiel finden Sie unter dem Namen **EDI-Order.mfd** im Ordner ...MapForceExamples.

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xsi:noNamespaceSchemaLocation="C:/DOCUME~1
http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://w
<Header>
 <Number>ABC123456XYZ</Number>
 <Received>2004-04-30T17:42:00-09:00</Received>
</Header>
<Customer>
 <Number>123</Number>
```



**to-duration**

Das Ergebnis ist der Zeitdauerwert, der aus der kodierten Quelle (F2380) stammt und den Formatcode (F2379) erhalten hat.



**to-time**

Das Ergebnis ist der Zeitwert, der aus der kodierten Quelle (F2380) stammt und den Formatcode (F2379) erhalten hat.

### 23.5.3 lang

Die lang-Bibliothek enthält eine Reihe von Funktionen, die bei Auswahl der Sprachen Java, C# oder C++ zur Verfügung stehen.

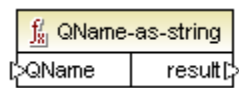
- [QName functions](#) (QName-Funktionen)
- [datetime functions](#) (Datums- und Uhrzeit-Funktionen)
- [generator functions](#) (Generator-Funktionen)
- [logical functions](#) (logische Funktionen)
- [math functions](#) (mathematische Funktionen)
- [string functions](#) (String-Funktionen)

#### QName functions (QName-Funktionen)



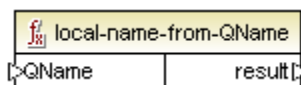
##### QName

Das Ergebnis ist ein QName, der anhand der Namespace URI und des lokalen Namens erstellt wird.



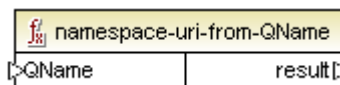
##### QName-as-string

Das Ergebnis ist eine eindeutige Darstellung des QName in Form eines String.



##### local-name-from-QName

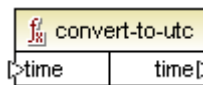
Das Ergebnis ist der lokale Namensteil des QName.



##### namespace-uri-from-QName

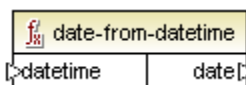
Das Ergebnis ist die Namespace URI des QName

#### datetime functions (Datums- und Uhrzeitfunktionen)



##### convert-to-utc

Konvertiert den lokalen "time"-Input-Parameter in UTC (Coordinated Universal Time) oder WEZ (Die Funktion berücksichtigt die Zeitzone-Komponente, z.B. +1:00)

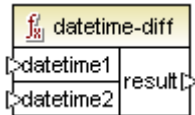


##### date-from-datetime

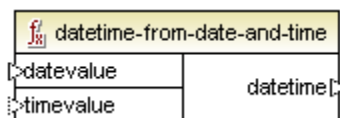
Das Ergebnis ist der Datumsteil eines Datums- und Uhrzeit Input-Arguments.

**datetime-add**

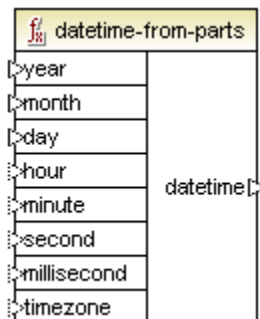
Das Ergebnis ist das Datum und die Uhrzeit, das/die durch Hinzufügen einer Zeitdauer (zweites Argument) zu einem Datum und einer Uhrzeit (erstes Argument) ermittelt wird/werden.

**datetime-diff**

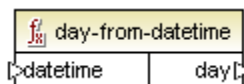
Das Ergebnis ist die Zeitdauer, die durch Subtraktion eines Datums und einer Uhrzeit (zweites Argument) von einem anderen Datum und einer anderen Uhrzeit (erstes Argument) ermittelt wird.

**datetime-from-date-and-time**

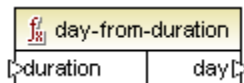
Das Ergebnis ist ein Datum und eine Uhrzeit, die aus einem Datumswert (erstes Argument) und einem Uhrzeitargument (zweites Argument) zusammengesetzt werden.

**datetime-from-parts**

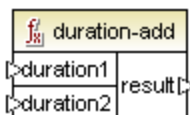
Das Ergebnis ist ein Datum und eine Uhrzeit, die aus jeder beliebigen Kombination der folgenden als Argumente bereitgestellten Teile zusammengesetzt werden: year, month, day, hour, minute, second, millisecond, timezone.

**day-from-datetime**

Das Ergebnis ist der durch das bereitgestellte Datums- und Uhrzeitargumente dargestellte Tag.

**day-from-duration**

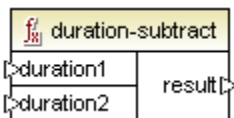
Das Ergebnis ist der durch das bereitgestellte Zeitdauerargument dargestellte Tag.

**duration-add**

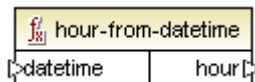
Das Ergebnis ist die durch Addition zweier Zeitdauerwerte ermittelte Zeitdauer.

**duration-from-parts**

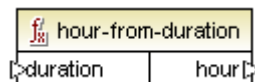
Das Ergebnis ist eine durch Kombination der folgenden als Argumente bereitgestellten Teile berechnete Zeitdauer: year, month, day, hour, minute, second, millisecond, negative.

**duration-subtract**

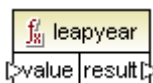
Das Ergebnis ist eine durch Subtraktion von duration2 von duration1 ermittelte Zeitdauer.

**hour-from-datetime**

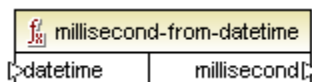
Das Ergebnis ist der Stundenteil des bereitgestellten Datums- und Uhrzeitarguments.

**hour-from-duration**

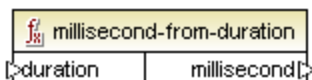
Das Ergebnis ist der Stundenteil des bereitgestellten Zeitdauerarguments.

**leapyear**

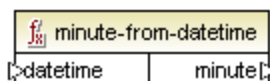
Das Ergebnis ist "true" oder "false", je nachdem, ob es sich bei dem bereitgestellten Datums- und Uhrzeitwert um ein Schaltjahr handelt.

**millisecond-from-datetime**

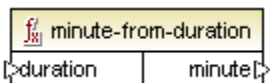
Das Ergebnis ist der Millisekundenteil des bereitgestellten Datums- und Uhrzeitarguments

**millisecond-from-duration**

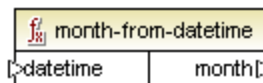
Das Ergebnis ist der Millisekundenteil des bereitgestellten Zeitdauerarguments.

**minute-from-datetime**

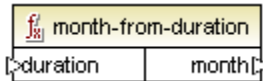
Das Ergebnis ist der Minutenteil des bereitgestellten Datums- und Uhrzeitarguments

**minute-from-duration**

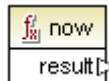
Das Ergebnis ist der Minutenteil des bereitgestellten Zeitdauerarguments.

**month-from-datetime**

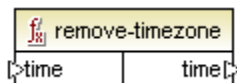
Das Ergebnis ist der Monatsteil des bereitgestellten Datums- und Uhrzeitarguments.

**month-from-duration**

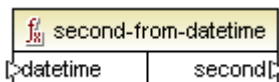
Das Ergebnis ist der Monatsteil des bereitgestellten Zeitdauerarguments.

**now**

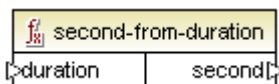
Das Ergebnis ist das aktuelle Datum und die aktuelle Uhrzeit (einschließlich der Zeitzone).

**remove-timezone**

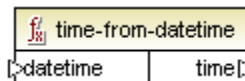
Entfernt die Zeitzonen-Komponente, z.B. +1:00 aus dem time-Input-Parameter.

**second-from-datetime**

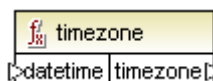
Das Ergebnis ist der Sekundenteil des bereitgestellten Datums- und Uhrzeitarguments.

**second-from-duration**

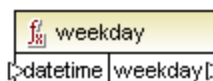
Das Ergebnis ist der Sekundenteil des bereitgestellten Zeitdauerarguments.

**time-from-datetime**

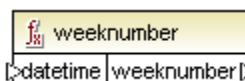
Das Ergebnis ist der Uhrzeitteil des bereitgestellten Datums- und Uhrzeitarguments.

**timezone**

Gibt die Zeitzone relativ zur UTC des bereitgestellten Datums- und Uhrzeitwerts zurück. Die Einheit, in der der Unterschied ausgegeben wird, sind Minuten.

**weekday**

Gibt den Wochentag des bereitgestellten Datums- und Uhrzeitwerts beginnend mit Monday=1 bis Sunday=7 zurück.

**weeknumber**

Gibt die Kalenderwoche, die den bereitgestellten Datums- und Uhrzeitwert dargestellt wird, zurück.



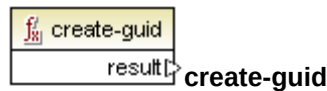
Das Ergebnis ist das Jahr des bereitgestellten Datums- und Uhrzeitwerts.



Das Ergebnis ist das Jahr des bereitgestellten Zeitdauerarguments.

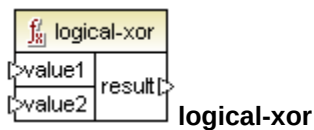
### generator functions (Generierungsfunktionen)

Die Generator-Funktionen generieren Werte für Datenbankfelder, die keine Input-Daten aus der Schema-, Datenbank- oder EDI-Quellkomponente haben.

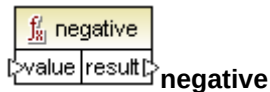


Das Ergebnis ist ein global eindeutiger Identifier (als hex-kodierter String) für das jeweilige Feld.

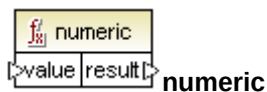
### logical functions (logische Funktionen)



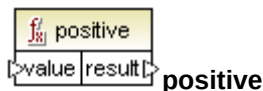
Das Ergebnis ist true, wenn value1 sich von value2 unterscheidet, andernfalls false.



Das Ergebnis ist true, wenn value negativ ist, d.h. kleiner als Null.

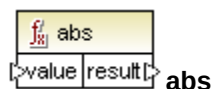


Das Ergebnis ist true, wenn value eine Zahl ist, andernfalls false. Beim Input-Wert handelt es sich normalerweise um einen String.



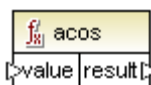
Das Ergebnis ist true, wenn value positiv ist, d.h. größer als Null, andernfalls false.

### math functions (mathematische Funktionen)



Das Ergebnis ist der absolute Wert des Input-Werts **value**.



**acos**

Das Ergebnis ist ArkusCosinus von **value**.

**asin**

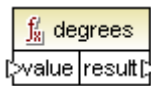
Das Ergebnis ist ArkusSinus von **value**.

**atan**

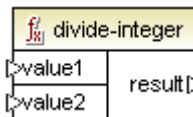
Das Ergebnis ist ArkusTangens von **value**.

**cos**

Das Ergebnis ist der Cosinus von **value**.

**degrees**

Das Ergebnis ist die Konvertierung von **value** von Bogenmaß in Grad.

**divide/integer**

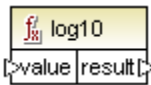
Das Ergebnis ist das Ganzzahlergebnis der Division von **value1** durch **value2**. z.B. ist bei 15 divide-integer 2 integer das Ganzzahlergebnis 7.

**exp**

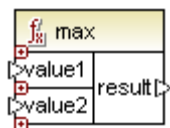
Das Ergebnis ist **e** (natürlicher Logarithmus) hoch **value**.

**log**

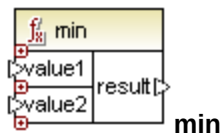
Das Ergebnis ist der natürliche Logarithmus von **value**.

**log10**

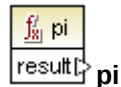
Das Ergebnis ist Logarithmus (Basis 10) von **value**.

**max**

Das Ergebnis ist der numerisch größere Wert von **value1** verglichen mit **value2**.



Das Ergebnis ist der numerisch kleinere Wert von **value1** verglichen mit **value2**.



Das Ergebnis ist der Wert von Pi.



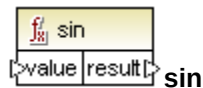
Das Ergebnis ist der Wert von **a** hoch **b**.



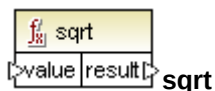
Das Ergebnis ist die Konvertierung von **value** in Bogenmaßgrade.



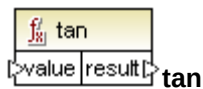
Das Ergebnis ist eine Zufallswert zwischen 0,0 und 1.0



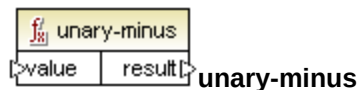
Das Ergebnis ist der Sinuswert von **value**.



Das Ergebnis ist die Quadratwurzel von **value**.

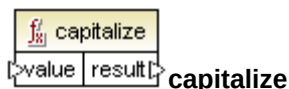


Das Ergebnis ist die Tangente von **value**.



Das Ergebnis ist der Umkehrwert des Input-Werts, z.B. ist das Ergebnis von +3 -3. -3 hat das Ergebnis +3.

### string functions (String-Funktionen)



Das Ergebnis ist der Input-String **value**, wobei der erste Buchstabe jedes Worts groß geschrieben ist.

| f <sub>32</sub> count-substring |         |
|---------------------------------|---------|
| >string                         | result< |
| >substr                         |         |

**count-substring**

Das Ergebnis ist die Anzahl, die dieser **substr.** (Substring) in **string** vorkommt.

| f <sub>33</sub> empty |         |
|-----------------------|---------|
| >value                | result< |

**empty**

Das Ergebnis ist true, wenn der Input-String **value** leer ist, andernfalls ist es false.

| f <sub>34</sub> find-substring |         |
|--------------------------------|---------|
| >string                        | result< |
| >substr                        |         |
| >startindex                    |         |

**find-substring**

Gibt die Position der ersten Instanz des substr. (Substring) im String an und zwar beginnend mit der Position startindex. Das erste Zeichen hat die Position 1. Wenn der Substring nicht gefunden wurde, ist das Ergebnis 0.

| f <sub>35</sub> format-guid-string |                 |
|------------------------------------|-----------------|
| >unformatted_guid                  | formatted_guid< |

**format-guid-string**

Das Ergebnis ist ein korrekt formatierter GUID String **formatted\_guid** für die Verwendung in Datenbankfeldern, wobei **unformatted\_guid** als Input-String in Datenbankfeldern verwendet wird. Siehe dazu auch [create\\_guid](#) Funktion unter generator functions.

| f <sub>36</sub> left |         |
|----------------------|---------|
| >string              | result< |
| >number              |         |

**left**

Das Ergebnis ist ein String, der die durch **number** definierte Anzahl der von links gezählten Zeichen des Input-String enthält. z.B: string="This is a sentence" und number=4, Ergebnis ist "This".

| f <sub>37</sub> left-trim |         |
|---------------------------|---------|
| >string                   | result< |

**left-trim**

Das Ergebnis ist der Input-String, wobei alle vorangestellten Whitespace-Zeichen entfernt wurden.

| f <sub>38</sub> lowercase |         |
|---------------------------|---------|
| >string                   | result< |

**lowercase**

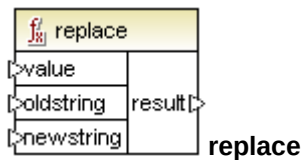
Das Ergebnis ist eine in Kleinbuchstaben geschriebene Version des Input String. Bei Unicode-Zeichen werden die entsprechenden (vom Unicode-Konsortium definierten) Kleinbuchstaben verwendet.

| f <sub>39</sub> match-pattern |         |
|-------------------------------|---------|
| >string                       | result< |
| >pattern                      |         |

**match-pattern**

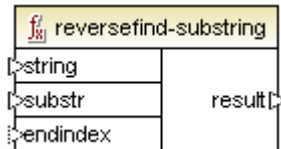
Das Ergebnis ist true, wenn der Input **String** dem durch **pattern** definierten regulären Ausdruck entspricht, andernfalls ist es false. Die Syntax des jeweiligen regulären Ausdrucks hängt von der Zielsprache ab.

Nähere Informationen zu Regular Expressions finden Sie unter [Regular Expressions](#).



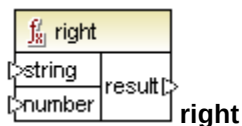
### replace

Das Ergebnis ist ein neuer String, wobei jede Instanz von **oldstring** im Input String **value** durch **newstring** ersetzt wird.



### reversefind-substring

Gibt die Position der ersten Instanz von **substr**, innerhalb von **string**, beginnend mit Position **endindex**, zurück, d.h. von rechts nach links. Das erste Zeichen hat die Position 1. Wenn der Substring nicht gefunden wurde, ist das Ergebnis 0.



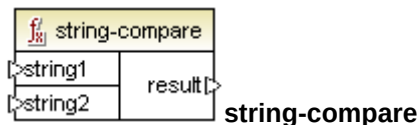
### right

Das Ergebnis ist ein String, der die durch **number** definierte Anzahl der von rechts gezählten Zeichen des Input-String enthält., z.B. string="This is a sentence" und number=5, Ergebnis ist "tence".



### right-trim

Das Ergebnis ist der Input-String, wobei alle nachfolgenden Whitespace-Zeichen entfernt wurden.



### string-compare

Gibt das Ergebnis eines String-Vergleichs von string1 mit string2 zurück, wobei die Groß- und Kleinschreibung berücksichtigt wird. Wenn string1=string2 dann ist das Ergebnis 0.

Wenn string1 kleiner als string2 ist, ist das Ergebnis < 0.

Wenn string1 größer als string2 ist, ist das Ergebnis > 0.

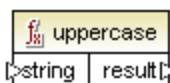


### string-compare-ignore-case

Gibt das Ergebnis eines String-Vergleichs von string1 mit string2 zurück, wobei die Groß- und Kleinschreibung ignoriert wird. Wenn string1=string2, dann ist das Ergebnis 0.

Wenn string1 kleiner als string2 ist, ist das Ergebnis < 0.

Wenn string1 größer als string2 ist, ist das Ergebnis > 0.

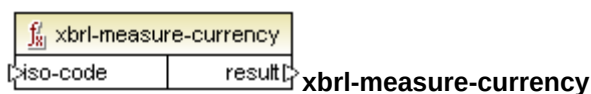
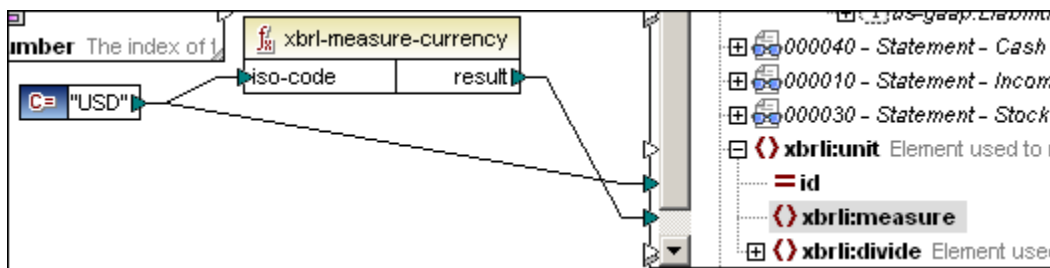
**uppercase**

Das Ergebnis ist eine in Großbuchstaben konvertierte Version des Input String. Bei Unicode-Zeichen werden die entsprechenden (vom Unicode-Konsortium definierten) Großbuchstaben verwendet.

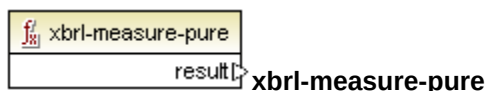
### 23.5.4 xbrl

Die XBRL-Bibliothek enthält Funktionen, die Daten in das für XBRL-Instanz-/Taxonomiedateien erforderliche QName-Format konvertieren.

Die Child-Elemente xbrl:ID und xbrl:measure des Elements xbrl:unit sind in einer XBRL-Instanzdatei obligatorisch und müssen gemappt werden, damit das Mapping gültig ist.



Das Ergebnis ist der QName für Datenelemente des Typs monetaryItemType aus dem ISO 4217-Währungscode.



Das Ergebnis ist der QName für Tarif-, Prozentsatz- oder Verhältnisdatenelemente.



Das Ergebnis ist der QName für Datenelemente vom Typ sharesItemType.

### 23.5.5 xlsx

Die XLSX-Bibliothek enthält Funktionen, die Daten von/in Excel Datums- und Uhrzeitformate konvertieren.

Beachten Sie beim Generieren von Code für C#, dass diese Funktionen von der Unterstützung für Excel-Komponenten abhängig ist, wofür .NET 3.0 oder höher benötigt wird. Daher muss unter **Extras | Optionen | Code-Generierung | C#-Einstellungen** Visual Studio 2005 / 2008 oder 2010 ausgewählt sein, damit diese Funktionen zur Verfügung stehen.

 columnname-to-index

 name  result

**columnname-to-index**

Das Ergebnis ist der Index der Spalte des angegebenen Namens. Spalte 1 erhält den Namen "A".

 date-to-xlsx

 date  result

**date-to-xlsx**

Das Ergebnis ist die Excel-Darstellung des gelieferten Datums- / Uhrzeit / Datum-und-Uhrzeitwerts.

 datetime-to-xlsx

 datetime  result

**datetime-to-xlsx**

Das Ergebnis ist die Excel-Darstellung des gelieferten Datum-und-Uhrzeitwerts.

 index-to-columnname

 n  result

**index-to-columnname**

Das Ergebnis ist der Name der Spalte "n", wobei die Spalte 1 "A" ist.

 time-to-xlsx

 time  result

**time-to-xlsx**

Das Ergebnis ist die Excel-Darstellung des gelieferten Uhrzeitwerts.

 xlsx-to-date

 number  result

**xlsx-to-date**

Das Ergebnis ist der Datumswert des Excel-Datenfelds.

 xlsx-to-datetime

 number  result

**xlsx-to-datetime**

Das Ergebnis ist der Datums- und Uhrzeitwert des Excel Felds für Datum und Uhrzeit.

 xlsx-to-time

 number  result

**xlsx-to-time**

Das Ergebnis ist der Uhrzeitwert des Excel Uhrzeitfeldes.

## 23.5.6 xpath2

XPath2-Funktionen stehen zur Verfügung, wenn entweder XSLT2 oder XQuery ausgewählt ist.

- [accessor functions \(accessor-Funktionen\)](#)
- [anyURIfunctions \(Alle URI-Funktionen\)](#)
- boolean functions (Boolesche Funktionen)
- constructors (Konstruktoren)
- context functions (Kontextfunktionen)
- durations, date and time functions (Zeitdauer-, Datums- und Uhrzeitfunktionen)
- node functions (Node-Funktionen)
- numeric functions (numerische Funktionen)
- [QName functions \(QName-Funktionen\)](#)
- string functions (String-Funktionen)

### accessors (Accessor-Funktionen)

Die folgenden accessor-Funktionen stehen zur Verfügung:

#### base-uri

Die `base-uri` Funktion erhält ein Node-Argument als Input und gibt die URI der XML-Ressource zurück, die den Node enthält. Die Ausgabe hat den Typ `xs:string`. MapForce gibt einen Fehler zurück, wenn kein Input-Node bereitgestellt wird.

Die Ausgabe hat den Typ `xs:string`.

#### document-uri

Nicht implementiert.

#### node-name

Die `node-name` Funktion erhält einen Node-Namen als Input-Argument und gibt seinen QName zurück. Wenn der QName als String dargestellt wird, hat er die Form `präfix:lokalerName`, wenn der Node ein Präfix hat, oder `lokalerName`, wenn der Node kein Präfix hat. Um die Namespace URI eines Node zu eruieren, verwenden Sie die `namespace-URI-from-QName` Funktion (in der Bibliothek der Funktionen im Zusammenhang mit QName).

#### string

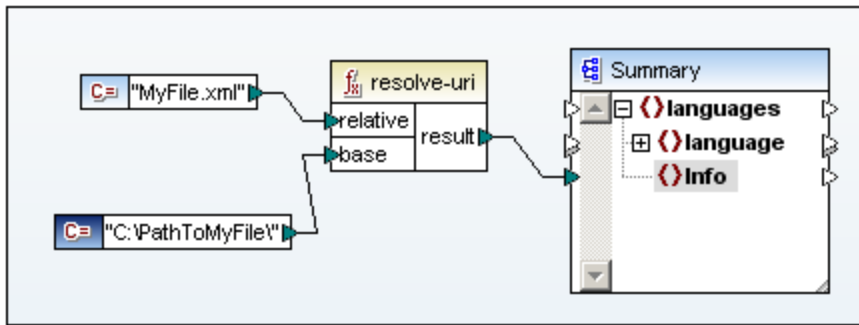
Die `string` Funktion funktioniert wie der `xs:string` Konstruktor: Sie konvertiert ihr Argument in `xs:string`. Wenn das Input-Argument ein Wert eines Typs "atomic" ist (z.B. `xs:decimal`), wird dieser Wert in einen Wert vom Typ `xs:string` konvertiert. Wenn das Input-Argument ein Node ist, wird der String-Wert des Node extrahiert. (Der String-Wert eines Node ist eine Verkettung der Werte der untergeordneten Werte des Node.)

### anyURI functions (Alle URI-Funktionen)

Die `resolve-uri` Funktion erhält eine URI als erstes Argument (Datentyp `xs:string`) und löst diese anhand der URI im zweiten Argument (Datentyp `xs:string`) auf.

Das Ergebnis (Datentyp `xs:string`) ist eine kombinierte URI. Auf diese Art kann eine relative URI (das erste Argument) durch Auflösung der URI anhand der Basis URI in eine absolute URI konvertiert werden.





In der Abbildung oben liefert das erste Argument die relative URI, das zweite Argument die Basis-URI. Die aufgelöste URI ist eine Verkettung der Basis-URI und der relativen URI, also `c:\PfadZuMeinerDatei\MeineDatei.xml`.

**Anmerkung:** Beide Argumente haben den Datentyp `xs:string`. Die Kombination erfolgt durch Behandlung beider Inputs als Strings. Es gibt daher keine Möglichkeit zu überprüfen, ob die durch diese URI definierten Ressourcen tatsächlich vorhanden sind. MapForce gibt einen Fehler zurück, wenn das zweite Argument fehlt.

### boolean functions (Boolesche Funktionen)

Die Booleschen Funktionen `true` und `false` erhalten kein Argument und geben die Booleschen Konstantenwerte `true` und `false` zurück. Sie können verwendet werden, wenn ein Boolescher Konstantenwert benötigt wird.

#### true

Fügt den Booleschen Wert "true" ein.

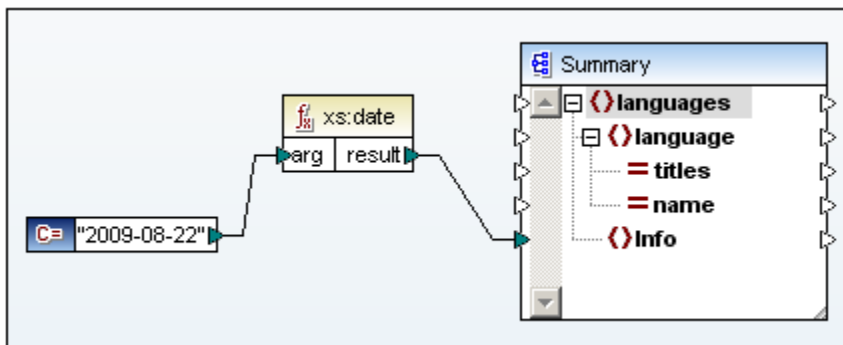
#### false

Fügt den Booleschen Wert "false" ein.

### constructors (Konstruktoren)

Die Funktionen in der XPath 2.0-Funktionsbibliothek unter der Überschrift "Constructors" konstruieren anhand des Input-Texts bestimmte Datentypen. Normalerweise muss der Input-Text das dem zu konstruierenden Datentyp entsprechende lexikalische Format haben, da die Transformation sonst nicht funktioniert.

Wenn Sie z.B. einen `xs:date` Datentyp erstellen möchten, verwenden Sie die `xs:date` Konstruktorfunktion. Der Input-Text muss das lexikalische Format des `xs:date` Datentyps haben, also `JJJJ-MM-TT` (Abbildung unten).



In der Abbildung oben wurde als Input-Argument der Funktion eine String-Konstante (2009-08-22) verwendet. Dieser Input hätte auch aus einem Node im Quelldokument abgerufen werden können.

Die `xs:date` Funktion gibt den Input-Text (2009-08-22), der den Datentyp `xs:string` (definiert in der *Constant* Komponente) hat, als Ausgabe mit dem Datentyp `xs:date` zurück.

Wenn Sie in einem Funktionskasten den Mauszeiger über das Input-Argument platzieren, wird der erwartete Datentyp des Arguments in einem Popup-Fenster angezeigt.

### context functions (Kontextfunktionen)

Die Bibliothek "context functions" enthält Funktionen, die das aktuelle Datum und die Uhrzeit bereitstellen, die Standard-Collation, die vom Prozessor verwendet wird, die Größe der aktuellen Sequenz und die Position des aktuellen Node.

#### Datums- und Uhrzeit-Funktionen

Die Funktionen `current-date`, `current-time` und `current-dateTime` erhalten kein Argument und geben das aktuelle Datum und/oder die aktuelle Uhrzeit der Systemuhr zurück.

Der Datentyp des Ergebnisses ist von der jeweiligen Funktion abhängig: `current-date` gibt `xs:date` zurück, `current-time` gibt `xs:time` zurück und `current-dateTime` gibt `xs:dateTime` zurück.

#### default-collation

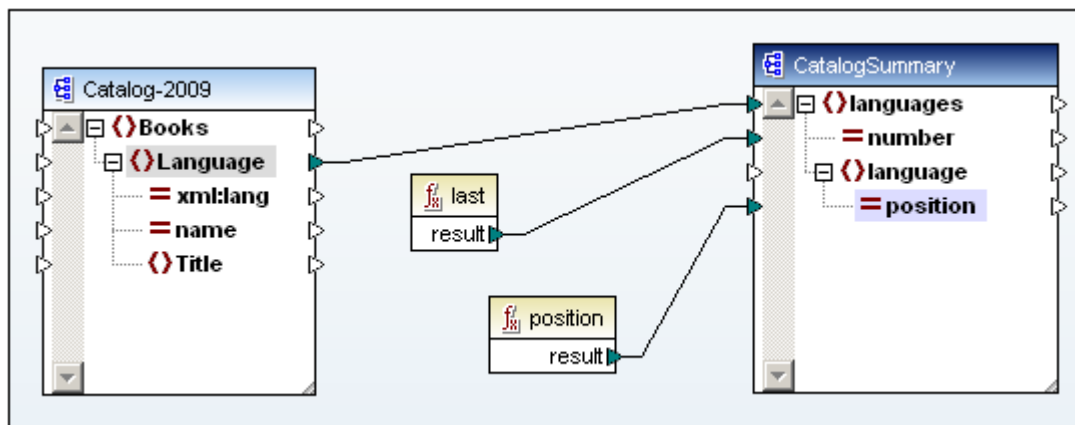
Die Funktion `default-collation` erhält kein Argument und gibt die Standard-Collation zurück, d.h. die Collation, die verwendet wird, wenn keine Collation für eine Funktion, für die eine solche definiert werden kann, festgelegt ist.

Der Altova XSLT 2.0 Prozessor unterstützt nur die Unicode Codepoint Collation. Vergleiche wie für die Funktionen `fn:max` und `fn:min` basieren auf dieser Collation.

#### last, position

Die Funktionen `last` und `position` erhalten kein Argument. Die `last` Funktion gibt die Position des letzten Node im Kontext-Nodeset zurück. Die `position` Funktion gibt die Position des aktuellen Node im Nodeset, das gerade verarbeitet wird, zurück.

Das Kontext-Nodeset bei den Nodes, an die die Funktion gerichtet ist, ist das Nodeset, auf das die Funktionen angewendet wird. In der Abbildung unten ist das Nodeset der Language Elemente das Kontext-Nodeset für die Funktionen `last` und `position`.



Im obigen Beispiel gibt die `last` Funktion die Position des letzten Node des Kontext-Nodeset (des Nodeset der Language Elemente) als den Wert des `number` Attributs zurück. Dieser Wert entspricht auch der Größe des Nodeset, da er anzeigt, wie viele Nodes im Nodeset enthalten sind.

Die `position` Funktion gibt die Position des Language Node, der gerade verarbeitet wird, zurück. Für jeden Language Element-Node wird die Position innerhalb des Nodes der Language Elemente im Attribut-Node `language/@position` ausgegeben.

Wir empfehlen Ihnen, die Funktionen **position** und **count** aus der **core** Bibliothek zu verwenden.

### **durations, date and time functions (Zeitdauer-, Datums- und Uhrzeitfunktionen)**

Mit Hilfe der Zeitdauer-, Datums- und Uhrzeitfunktionen können Sie das Datum und die Uhrzeit für die Zeitzone anpassen, bestimmte Komponenten aus Datums- und Uhrzeitdaten extrahieren und eine Datums- und Uhrzeit-Einheit von einer anderen subtrahieren.

#### **Die 'Adjust-to-Timezone' Funktionen**

Jede dieser Funktionen erhält ein Datum, eine Uhrzeit oder ein Datum- und eine Uhrzeit als erstes Argument und passt die Input-Daten je nach dem Wert des zweiten Arguments durch Hinzufügen, Entfernen oder Ändern der Zeitzonen-Komponente an.

Wenn das erste Argument keine Zeitzone enthält (z.B. das Datum `2009-01` oder die Uhrzeit `14:00:00`), ergeben sich die folgenden Möglichkeiten:

- Das Argument "timezone" (das zweite Argument in der Funktion) ist vorhanden: Das Ergebnis enthält die im zweiten Argument definierte Zeitzone. Die Zeitzone im zweiten Argument wird hinzugefügt.
- Das Argument "timezone" (das zweite Argument in der Funktion) fehlt: Das Ergebnis enthält die implizite Zeitzone, also die Zeitzone des Systems. Die Zeitzone des Systems wird hinzugefügt.
- Das Argument "timezone" (das zweite Argument in der Funktion) ist leer: Das Ergebnis enthält keine Zeitzone.

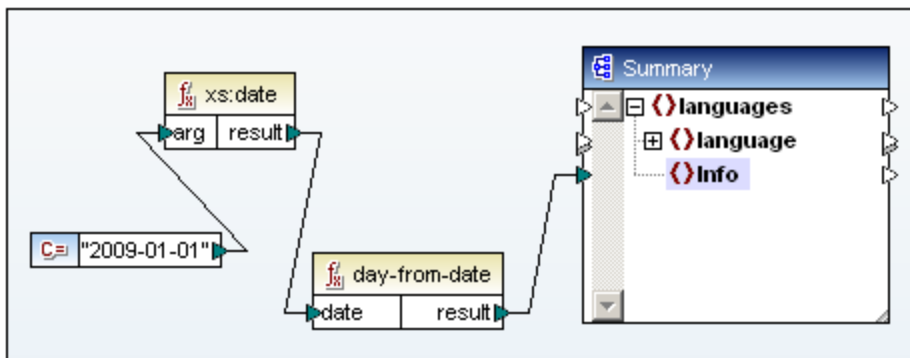
Wenn das erste Argument eine Zeitzone enthält, (z.B. das Datum `2009-01-01+01:00` oder die Uhrzeit `14:00:00+01:00`), ergeben sich die folgenden Möglichkeiten:

- Das Argument "timezone" (das zweite Argument in der Funktion) ist vorhanden: Das Ergebnis enthält die im zweiten Argument definierte Zeitzone. Die ursprüngliche Zeitzone wird durch die im zweiten Argument definierte Zeitzone ersetzt.
- Das Argument "timezone" (das zweite Argument in der Funktion) fehlt: Das Ergebnis enthält die implizite Zeitzone, also die Zeitzone des Systems. Die ursprüngliche Zeitzone wird durch die Zeitzone des Systems ersetzt.
- Das Argument "timezone" (das zweite Argument in der Funktion) ist leer: Das Ergebnis enthält keine Zeitzone.

#### **Die 'From' Funktionen**

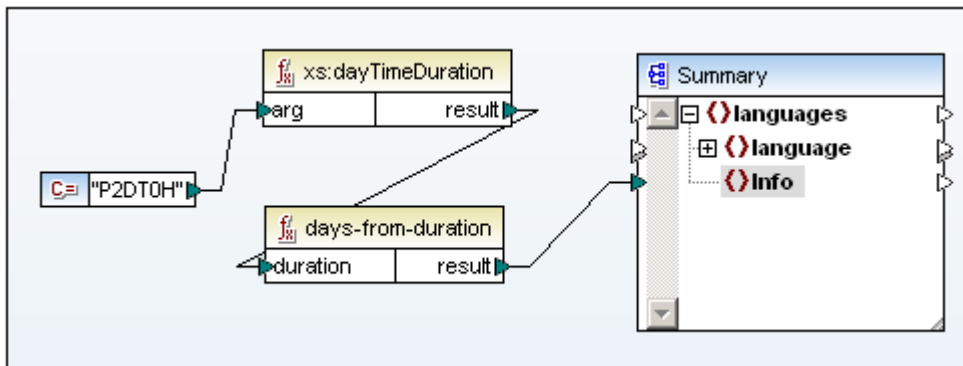
Die einzelnen 'From' Funktionen extrahieren eine bestimmte Komponente aus: (i) Datums- oder Uhrzeitdaten, und (ii) Zeitdauerdaten. Die Ergebnisse haben den Datentyp `xs:decimal`.

Als Beispiel für die Extraktion einer Komponente aus Datums- oder Uhrzeitdaten wird hier die `day-from-date` Funktion verwendet (*Abbildung unten*).



Das Input-Argument ist ein Datum (2009-01-01) vom Typ xs:date. Die day-from-date Funktion extrahiert die Tageskomponente des Datums (1) als einen xs:decimal Datentyp.

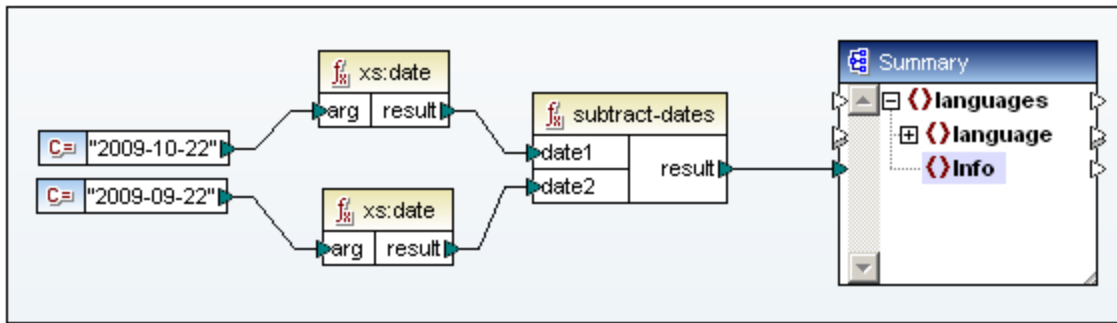
Wenn Zeitkomponenten aus einer Zeitdauer extrahiert werden, muss die Zeitdauer entweder als xs:yearMonthDuration (zum Extrahieren von Jahren und Monaten) oder als xs:dayTimeDuration (zum Extrahieren von Tagen, Stunden, Minuten und Sekunden) definiert werden. Das Ergebnis hat den Typ xs:decimal. In der Abbildung unten sehen Sie dayTimeDuration von P2DT0H als Input für die Funktion days-from-duration. Das Ergebnis ist xs:decimal Datentyp 2.



### Die 'Subtract' Funktionen

Mit Hilfe der Subtraktionsfunktionen können Sie einen Uhrzeitwert von einem anderen subtrahieren und einen Zeitdauerwert errechnen. Die drei Subtraktionsfunktionen sind: subtract-dates, subtract-times, subtract-dateTimes.

In der Abbildung unten sehen Sie, wie ein Datum mit Hilfe der subtract-dates-Funktion von einem anderen Datum subtrahiert wird (2009-10-22 minus 2009-09-22). Das Ergebnis ist der dayTimeDuration Wert P30D.



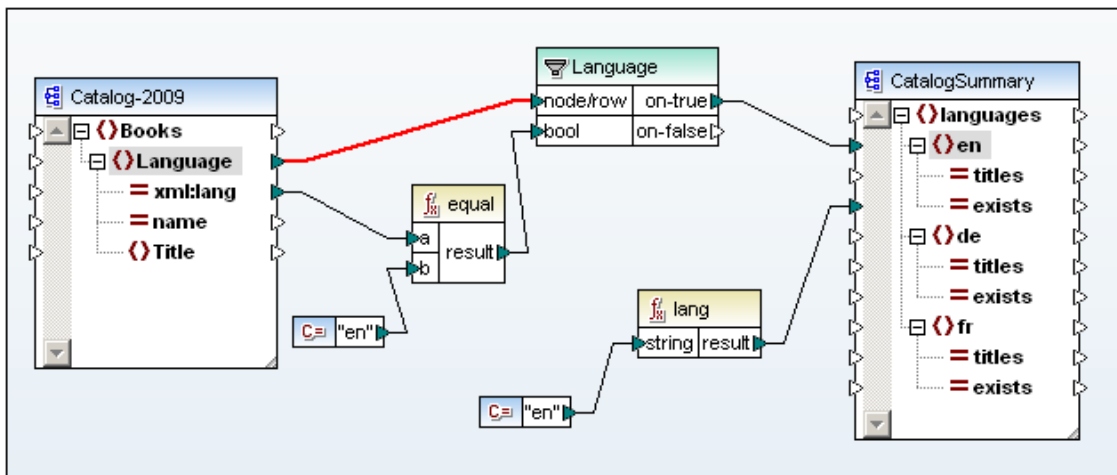
**Anmerkung:** Wenn Sie den Mauszeiger in einem Funktionskasten über das Input-Argument platzieren, wird der erwartete Datentyp in einem Popup-Fenster angezeigt.

### node functions (Node-Funktionen)

Die folgenden Node-Funktionen stehen zur Verfügung:

#### lang

Die lang Funktion erhält ein String-Argument, das einen Sprachcode definiert (wie z.B. en). Abhängig davon, ob der Kontext-Node ein `xml:lang` Attribut mit einem Wert hat, der mit dem Argument der Funktion übereinstimmt, wird entweder `true` oder `false` ausgegeben.



Beachten Sie in der Abbildung oben Folgendes:

1. Im Quellschema hat das Element `Language` ein `xml:lang` Attribut.
2. `Language` Nodes werden gefiltert, sodass nur jene `Language` Nodes mit dem `xml:lang` Wert `en` verarbeitet werden (der Filter ist in der `equal` Funktion definiert).
3. Der `Language` Node ist der Kontext-Node an der Stelle, an der im Ausgabedokument das Element `en` erstellt wird.
4. Die Ausgabe der `lang` Funktion (`true` oder `false`) wird an den `en/@exists` Attribut-Node der Ausgabe gesendet. Das Argument der Funktion wird durch die String-Konstante `en` bereitgestellt. Die `lang` Funktion überprüft anschließend, ob der Kontext-Node an dieser Stelle (dem `Language` Element) ein `xml:lang` Attribut mit dem Wert `en` (dem Argument der Funktion) hat. Wenn ja, so wird `true` zurückgegeben, andernfalls `false`.

#### local-name, name, namespace-uri

Die Funktionen `local-name`, `name` und `namespace-uri` geben den lokalen Namen, den Namen

bzw. die Namespace URI des Input-Node zurück. So ist z.B. beim Node `altova:Products` der lokale Name `Products`, der Name `altova:Products` und die Namespace URI ist die URI des Namespace an den das Präfix `altova:` gebunden ist (z.B. `http://www.altova.com/examples`).

Jeder dieser drei Funktionen hat zwei Varianten:

- ohne Argument: In diesem Fall wird die Funktion auf den Kontext-Node (ein Beispiel für einen Kontext-Node finden Sie im Beispiel oben zur `lang` Funktion) angewendet.
- ein Argument, das ein Node sein muss: Die Funktion wird auf den angegebenen Node angewendet.

Das Ergebnis dieser insgesamt sechs Varianten ist ein String.

### **number**

Konvertiert einen Input-String in eine Zahl. Konvertiert auch einen Booleschen Input in eine Zahl.

Die `number`-Funktion erhält einen Node als Input, zerlegt den Node in seine Bestandteile (d.h. extrahiert seinen Inhalt), konvertiert den Wert in eine Dezimalzahl und gibt den konvertierten Wert zurück. Die einzigen Typen, die in Zahlen konvertiert werden können, sind Boolesche Werte, Strings und andere numerische Typen. Nicht numerische Input-Wert (wie z.B. ein nicht numerischer String) führen zum Ergebnis `NaN` (Not a Number).

Die `number`-Funktion hat zwei Varianten:

- ohne Argument: In diesem Fall wird die Funktion auf den Kontext-Node (ein Beispiel für einen Kontext-Node finden Sie im Beispiel oben zur `lang` Funktion) angewendet.
- ein Argument, das ein Node sein muss: Die Funktion wird auf den angegebenen Node angewendet.

### **numeric functions (numerische Funktionen)**

Die folgenden numerischen Funktionen stehen zur Verfügung:

#### **abs**

Die `abs` Funktion erhält einen numerischen Wert als Input und gibt seinen absoluten Wert als Dezimalzahl zurück. Wenn das Input-Argument z.B. `-2` oder `+2` ist, so gibt die Funktion den Wert `2` zurück.

#### **round-half-to-even**

Die `round-half-to-even` Funktion rundet die bereitgestellte Zahl (das erste Argument) auf den Präzisionsgrad (Anzahl der Dezimalstellen) auf bzw. ab, der im optionalen zweiten Argument definiert ist. Wenn z.B. das erste Argument `2,141567` und das zweite Argument `3` ist, dann wird das erste Argument (die Zahl) auf drei Dezimalstellen gerundet, d.h. das Ergebnis ist `2,141`. Wenn kein Präzisionsgrad (zweites Argument) angegeben ist, wird die Zahl auf null Dezimalstellen, also eine Ganzzahl gerundet.

'even' im Funktionsnamen bezieht sich auf die Rundung auf eine gerade Zahl, wenn eine Ziffer in einer Zahl sich genau in der Mitte zwischen zwei Werten befindet. `round-half-to-even` (`3,475, 2`) ergäbe z.B. `3,48`.

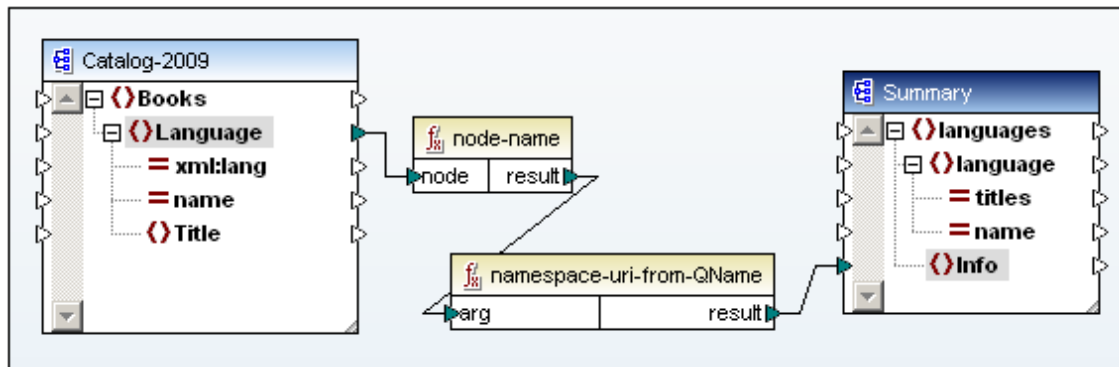
### qname-related functions (QName-Funktionen)

Es gibt zwei QName-Funktionen, die ähnlich funktionieren: `local-name-from-QName` und `namespace-uri-from-QName`.

Beide Funktionen erhalten einen erweiterten QName (in Form eines String) als Input-Argument und erzeugen den lokalen Namen bzw. den Namespace-URI Teil des erweiterten QName.

Beachten Sie bitte: Da die Input-Daten der beiden Funktionen Strings sind, kann ein Node nicht direkt mit den Input-Argument-Kästen dieser Funktionen verbunden werden.

Der Node muss zuerst an die `node-name` Funktion geliefert werden, die dann den erweiterten QName ausgibt. Dieser erweiterte QName kann dann als Input für die beiden Funktionen verwendet werden (siehe Abbildung unten).



Die Ausgabe der beiden Funktionen ist ein String.

### string functions (String-Funktionen)

Die folgenden String-Funktionen stehen zur Verfügung:

#### compare

Die `compare` Funktion erhält zwei Strings als Argumente und vergleicht diese alphabetisch und überprüft, ob diese identisch sind. Wenn *String-1* im Alphabet vor *String-2* (z.B. bei zwei Strings A und B) vorkommt, dann gibt die Funktion -1 zurück. Wenn die beiden Strings gleich sind (z.B. A und A), gibt die Funktion 0 zurück. Wenn *String-1* im Alphabet nach *String-2* (z.B. B und A) vorkommt, dann gibt die Funktion +1 zurück.

Eine Variante dieser Funktion ermöglicht Ihnen, auszuwählen, welche Collation zum Vergleichen der Strings verwendet werden soll. Wenn keine Collation verwendet wird, wird die Standard-Collation, nämlich die Unicode Codepoint Collation, verwendet. Der Altova Prozessor unterstützt nur die Codepoint Collation.

#### ends-with

Die `ends-with` Funktion überprüft, ob *String-1* mit *String-2* endet. Wenn ja, gibt die Funktion true zurück, andernfalls false.

Eine Variante dieser Funktion ermöglicht Ihnen, auszuwählen, welche Collation zum Vergleichen der Strings verwendet werden soll. Wenn keine Collation verwendet wird, wird die Standard-Collation, nämlich die Unicode Codepoint Collation verwendet. Der Altova Prozessor unterstützt nur die Codepoint Collation.

**escape-uri**

Die `escape-uri` Funktion erhält eine URI als Input für das erste String-Argument und wendet die URI Escape-Konventionen von RFC 2396 auf den String an. Das zweite Boolesche Argument (`escape-reserved`) sollte auf `true()` gesetzt werden, wenn Zeichen mit einer bereits reservierten Bedeutung in URIs mit Escape ausgelassen werden sollen (z.B. "+" oder "/").

Beispiel:

```
escape-uri("My A+B.doc", true()) würde My%20A%2B.doc zurückgeben
escape-uri("My A+B.doc", false()) würde My%20A+B.doc zurückgeben
```

**lower-case**

Die `lower-case` Funktion erhält einen String als Argument und konvertiert alle Großbuchstaben im String in die entsprechenden Kleinbuchstaben.

**matches**

Die `matches` Funktion überprüft, ob ein bereitgestellter String (das erste Argument) einem regulären Ausdruck (dem zweiten Argument) entspricht. Die Syntax des regulären Ausdrucks muss die Syntax sein, die für die `pattern` Facet von XML-Schema definiert wurde. Die Funktion gibt `true` zurück, wenn der String dem regulären Ausdruck entspricht, und andernfalls `false`.

Die Funktion erhält ein optionales `flags` Argument. Es sind vier Flags definiert (`i`, `m`, `s`, `x`). Es können mehrere Flags verwendet werden: z.B: `imx`. Wenn kein Flag verwendet wird, werden die Standardwerte aller vier Flags verwendet.

Die vier Flags haben die folgende Bedeutung:

- `i` Modus "Groß/Kleinschreibung wird nicht berücksichtigt" verwenden. Die Standardeinstellung ist Groß-/Kleinschreibung berücksichtigen.
- `m` Mehrzeiligen Modus verwenden. In diesem Modus wird der String als mehrzeilig betrachtet, wobei jede Zeile durch ein newline-Zeichen (`\n`) getrennt wird. Die Metazeichen `^` und `$` kennzeichnen den Beginn und das Ende der einzelnen Zeilen. Die Standardeinstellung ist der String-Modus, in dem der String mit den Metazeichen `^` und `$` endet.
- `s` dot-all Modus verwenden. Die Standardeinstellung ist der not-dot-all Modus, in dem das Metazeichen `.` für alle Zeichen mit Ausnahme des newline-Zeichens (`\n`) steht. Im dot-all Modus steht der Punkt auch für das newline-Zeichen.
- `x` Whitespaces ignorieren. Standardmäßig werden Whitespace-Zeichen nicht ignoriert.

**normalize-unicode**

Die `normalize-unicode` Funktion normalisiert den Input-String (das erste Argument) entsprechend den Regeln des definierten Normalisierungsprotokolls (zweites Argument). Unterstützt werden die Normalisierungsprotokolle NFC, NFD, NFKC und NFKD.

**replace**

Die `replace` Funktion erhält den im ersten Argument definierten String als Input, sucht nach den in einem regulären Ausdruck definierten Übereinstimmungen (zweites Argument) und ersetzt diese durch den String im dritten Argument.



Die Regeln zum Suchen von Übereinstimmungen sind im `matches`-Attribut oben definiert. Die Funktion erhält außerdem ein optionales `flags` Argument. Die Flags sind weiter oben in der `matches` Funktion beschrieben.

**starts-with**

Die `starts-with` Funktion überprüft, ob *String-1* mit *String-2*. beginnt. Falls ja, gibt die Funktion `true` zurück, andernfalls `false`.

In einer Variante dieser Funktion können Sie auswählen, welche Collation zum Vergleichen der Strings verwendet wird. Wenn keine Collation angegeben wird, wird die Standard-Collation, also die Unicode Codepoint Collation verwendet. Der Altova Prozessor unterstützt nur die Unicode Codepoint Collation.

**substring-after**

Die `substring-after` Funktion gibt den Teil von *String-1* (dem ersten Argument) zurück, der sich hinter dem Test-String *String-2* (dem zweiten Argument) befindet. Ein optionales drittes Argument definiert, welche Collation beim String-Vergleich verwendet werden soll. Wenn keine Collation angegeben wird, wird die Standard-Collation, also die Unicode Codepoint Collation verwendet. Der Altova Prozessor unterstützt nur die Unicode Codepoint Collation.

**substring-before**

Die `substring-before` Funktion gibt den Teil von *String-1* (dem ersten Argument) zurück, der sich vor dem Test-String *String-2* (dem zweiten Argument) befindet. Ein optionales drittes Argument definiert, welche Collation beim String-Vergleich verwendet werden soll. Wenn keine Collation angegeben wird, wird die Standard-Collation, also die Unicode Codepoint Collation verwendet. Der Altova Prozessor unterstützt nur die Unicode Codepoint Collation.

**upper-case**

Die `upper-case` Funktion erhält einen String als Argument und konvertiert alle Kleinbuchstaben im String in die entsprechenden Großbuchstaben.

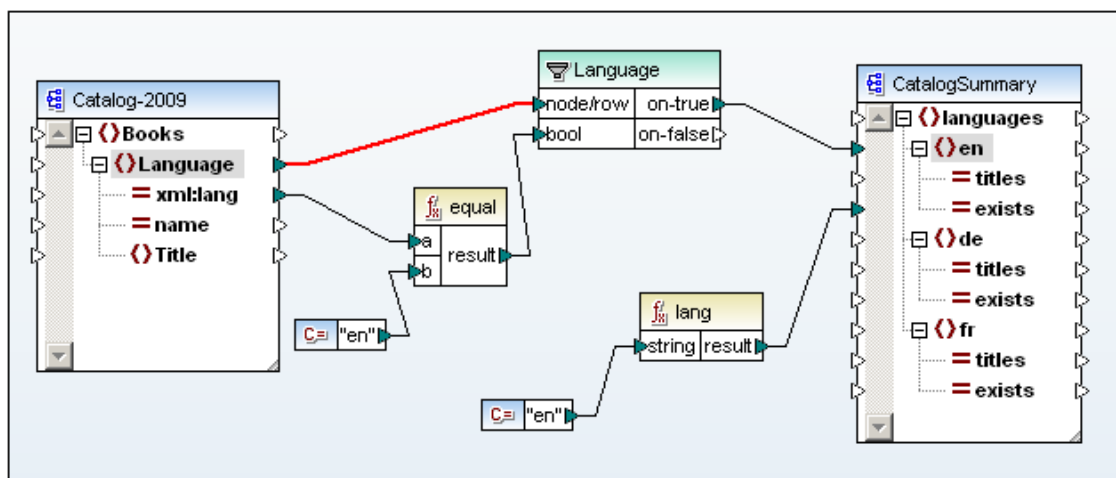
### 23.5.7 xslt

- [xpath functions](#)  
Die Funktionen in der XPath-Funktionsbibliothek sind XPath 1.0 Nodeset-Funktionen.
- [xslt functions](#)  
Die Funktionen in der XSLT-Funktionsbibliothek sind XSLT 1.0 Funktionen.

#### xpath functions (XPath-Funktionen)

Die Funktionen in der XPath-Funktionsbibliothek sind XPath 1.0 Nodeset-Funktionen. Jede dieser Funktionen erhält einen Node oder ein Nodeset als Kontext und gibt Informationen über diesen Node oder dieses Nodeset zurück. Diese Funktionen haben normalerweise:

- einen Kontext-Node (in der Abbildung unten ist der Kontext-Node für die lang Funktion das Element "Language" des Quellschemas).
- ein Input-Argument (in der Abbildung unten ist das Input-Argument für die lang Funktion die String-Konstante en). Die Funktionen last und position erhalten kein Argument.



#### lang

Die lang Funktion erhält ein String-Argument, das einen Sprachcode definiert (wie z.B. en). Die Funktion gibt je nachdem, ob der Kontext-Node ein `xml:lang` Attribut mit einem Wert hat, der mit dem Argument der Funktion übereinstimmt, entweder `true` oder `false` zurück. Beachten Sie in der Abbildung oben bitte Folgendes:

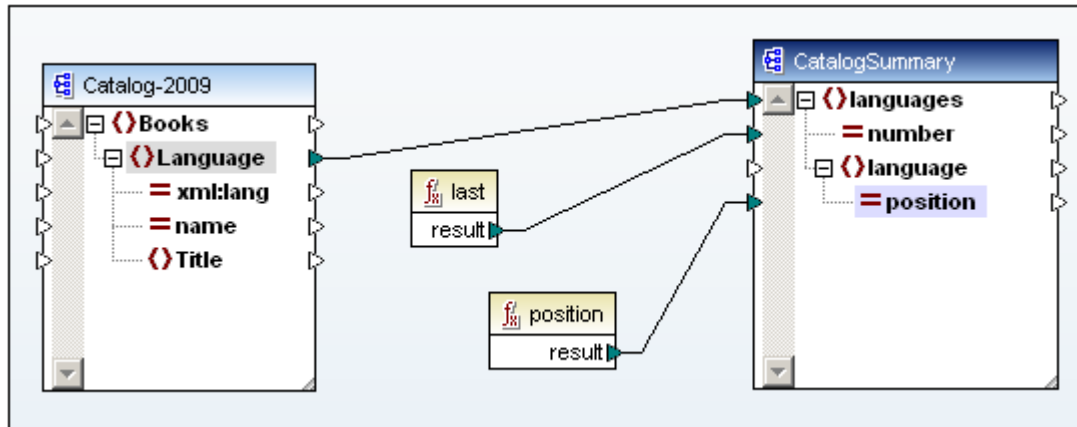
1. Das Language Element im Quellschema hat ein `xml:lang` Attribut.
2. Language Nodes werden gefiltert, sodass nur diejenigen Language Nodes mit dem `xml:lang` Wert en verarbeitet werden (der Filtertest ist in der `equal` Funktion definiert).
3. Der Language Node ist der Kontext-Node an der Stelle, an der im Ausgabedokument ein en Element erstellt wird.
4. Der Ausgabewert der lang Funktion (`true` oder `false`) wird an den `en/@exists` Attribut-Node der Ausgabe gesendet. Das Argument der Funktion wird durch die String-Konstante en bereitgestellt. Anschließend überprüft die lang Funktion, ob der Kontext-Node an dieser Stelle (dem Language Element) ein `xml:lang` Attribut mit dem Wert en (dem Argument der Funktion) hat. Falls ja, wird `true` zurückgegeben, andernfalls `false`.

#### last, position

Die Funktionen last und position erhalten kein Argument. Die last Funktion gibt die Position

des letzten Node im Kontext-Nodeset zurück. Die `position` Funktion gibt die Position des aktuellen Node im gerade verarbeiteten Nodeset zurück.

Das Kontext-Nodeset bei den Nodes, an die die Funktionen gerichtet sind, ist das Nodeset, auf das die Funktionen angewendet werden. In der Abbildung unten ist das Nodeset von Language Elementen das Kontext-Nodeset für die Funktionen `last` und `position`.



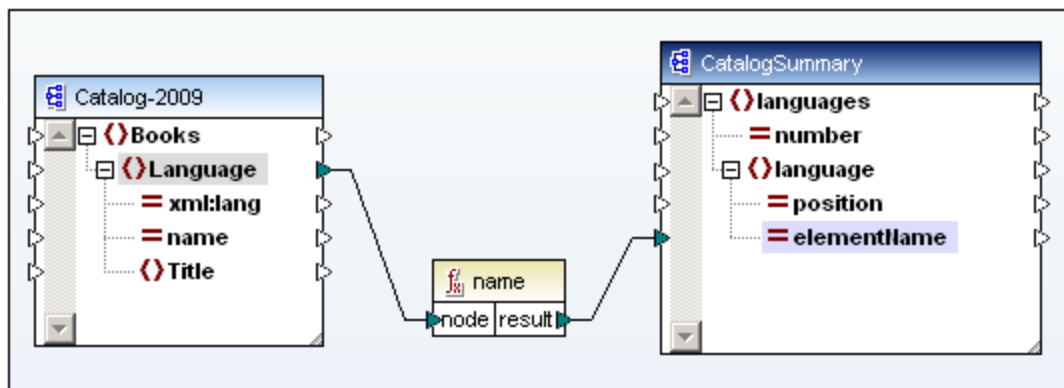
Im Beispiel oben gibt die `last` die Position des letzten Node des Kontext-Nodeset (des Nodeset der Language Elemente) als den Wert des `number` Attributs zurück. Dieser Wert entspricht auch der Größe des Nodeset, da er angibt, wie viele Nodes das Nodeset enthält.

Die `position` Funktion gibt die Position des gerade verarbeiteten Language Node zurück. Für jeden Language Element-Node wird die Position innerhalb des Nodeset der Language Elemente im `language/@position` Attribute-Node ausgegeben.

### name, local-name, namespace-uri

Diese Funktionen werden alle auf dieselbe Art verwendet. Sie geben den Namen, den lokalen Namen bzw. die Namespace URI des Input-Node zurück. In der Abbildung unten sehen Sie, wie diese Funktionen verwendet werden. Beachten Sie, dass kein Kontext-Node definiert ist.

Die `name`-Funktion gibt den Namen des Language Node zurück und gibt diesen an das `language/@elementname` Attribut aus. Wenn das Argument einer dieser Funktionen ein Nodeset anstelle eines einzelnen Node ist, wird der Name (bzw. der lokale Name oder die Namespace URI) des ersten Node im Nodeset zurückgegeben.



Die `name` Funktion gibt den QName des Node zurück; die `local-name` Funktion gibt den lokalen Teil des Names des QName zurück. Wenn der QName eines Node z.B. `altova:MyNode` ist,

dann ist MyNode der lokale Name.

Die Namespace URI ist die URI des Namespace, dem der Node angehört. Das Präfix `altova:` kann z.B. so deklariert werden, dass es der folgenden Namespace URI zugeordnet wird:  
`xmlns:altova="http://www.altova.com/namespaces"`.

**Anmerkung:** Weitere XPath 1.0 Funktionen finden Sie auch in der "Core" Funktionsbibliothek.

### xslt functions (XSLT-Funktionen)

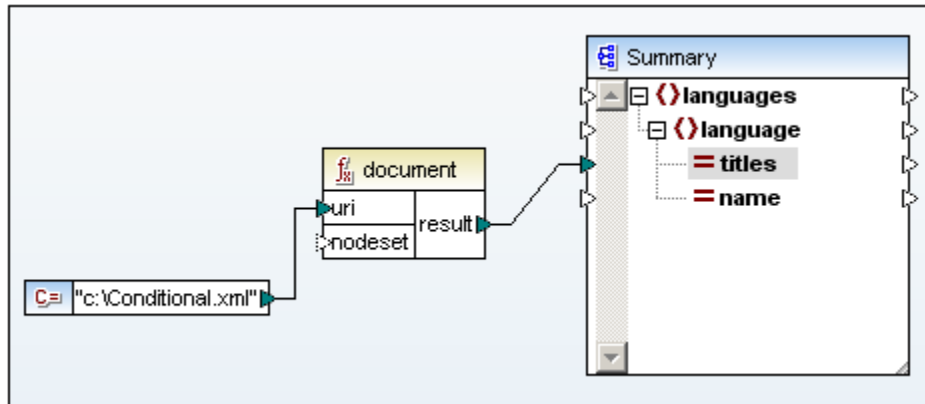
Die Funktionen in der XSLT-Funktionsbibliothek sind XSLT 1.0 Funktionen und werden weiter unten beschrieben. Um eine Funktion zu verwenden, ziehen Sie sie in das Mapping. Wenn Sie den Mauszeiger über den Input-Argumentteil eines Funktionskastens ziehen, wird der erwartete Datentyp des Arguments in einem Popup-Fenster angezeigt.

#### current

Die `current`-Funktion erhält kein Argument und gibt den aktuellen Node zurück.

#### document

Die `document` Funktion richtet sich (mit dem Argument `uri`; siehe Abbildung unten) an ein externes XML-Dokument. Das optionale `Nodeset`-Argument definiert einen Node, anhand dessen Basis-URI die URI aufgelöst wird, die als erstes Argument bereitgestellt wird, wenn diese URI relativ ist. Das Ergebnis wird in einen Node im Ausgabedokument ausgegeben.

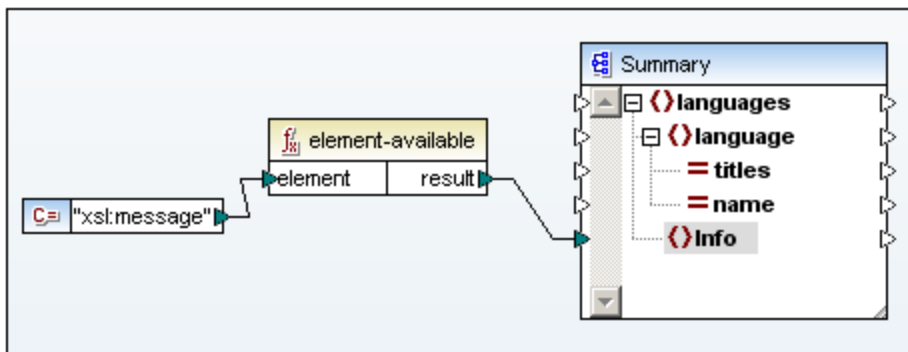


Beachten Sie, dass das `uri` Argument ein String ist, der ein absoluter Pfad sein muss.

#### element-available

Die `element-available` Funktion überprüft, ob ein Element, das als das einzige String-Argument der Funktion bereitgestellt wird, vom XSLT-Prozessor unterstützt wird.

Der Argument-String wird als QName ausgewertet. Daher müssen XSLT-Elemente ein `xsl:` Präfix und XML-Schema-Elemente ein `xs:` Präfix haben, da dies die Präfixe sind, die im zugrundeliegenden XSLT-Dokument, das für das Mapping erstellt wird, deklariert sind.



Die Funktion gibt einen Booleschen Wert zurück.

### function-available

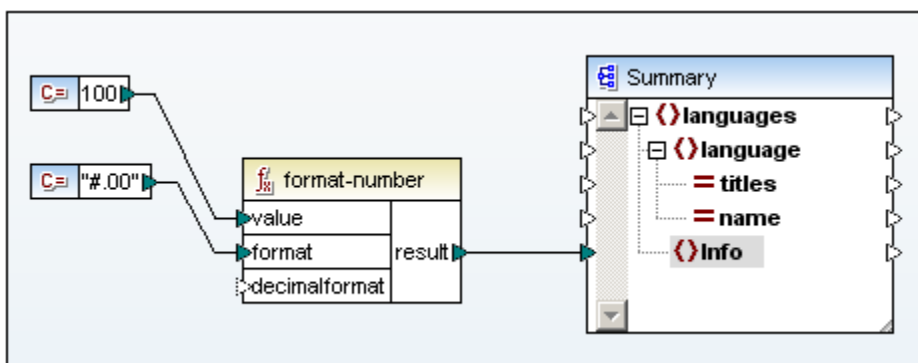
Die function-available Funktion ähnelt der element-available Funktion und überprüft, ob der als Argument der Funktion bereitgestellte Funktionsname vom XSLT-Prozessor unterstützt wird.

Der Input-String wird als QName ausgewertet. Die Funktion gibt einen Booleschen Wert zurück.

### format-number

Die format-number Funktion erhält eine Ganzzahl als erstes Argument (value) und einen Format-String als zweites Argument (format). Das dritte optionale Argument ist ein String, der angibt, welches Dezimalformat verwendet werden soll. Wenn dieses Argument nicht verwendet wird, wird das Standarddezimalformat verwendet.

Dezimalformate werden durch das XSLT 1.0 decimal-format Element definiert: Jedes so definierte Dezimalformat kann benannt werden und dieser Name kann als drittes Argument der format-number Funktion verwendet werden. Wenn ein Dezimalformat ohne Namen definiert wird, ist es das Standarddezimalformat für die Transformation.



Die Funktion gibt die als String formatierte Zahl zurück.

### generate-id

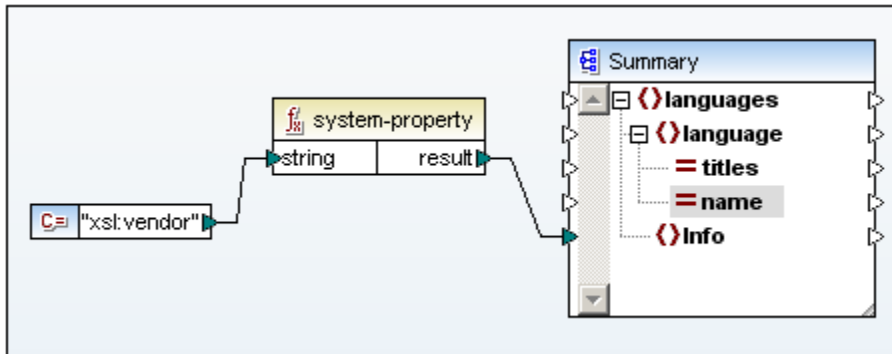
Die generate-id Funktion generiert einen eindeutigen String, der den ersten Node des Nodeset anhand des optionalen Input-Arguments identifiziert.

Wenn kein Argument bereitgestellt wird, wird die ID am Kontext-Node generiert. Das Ergebnis kann an jeden Node im Ausgabedokument gerichtet werden.

**system-property**

Die `system-property` Funktion gibt die Eigenschaften des XSLT-Prozessors (des Systems) zurück. Drei Systemeigenschaften, alle im XSLT-Namespace, sind bei XSLT-Prozessoren obligatorisch, nämlich `xsl:version`, `xsl:vendor` und `xsl:vendor-url`.

Der Input-String wird als QName ausgewertet und muss daher das `xsl:prefix` haben, da dies das Präfix ist, das im zugrundeliegenden XSLT-Stylesheet mit dem XSLT-Namespace verknüpft ist.

**unparsed-entity-uri**

Wenn Sie eine DTD verwenden, können Sie dafür eine ungeparste Entity deklarieren. Diese ungeparste Entity, z.B. ein Bild, hat eine URI, die den Pfad zur Entity angibt.

Der Input-String der Funktion muss dem Namen der in der DTD deklarierten ungeparsten Entity entsprechen, dann gibt die Funktion die URI der ungeparsten Entity zurück. Diese URI kann dann an einen Node im Ausgabedokument gerichtet werden, z.B. an einen `href` Node.

# Kapitel 24

---

## Implementieren von Webservices

## 24 Implementieren von Webservices

In MapForce können Sie anhand einer vorhandenen WSDL 1.1 oder 2.0-Datei, die Sie z.B. in XMLSpy erstellt haben, SOAP Webservices generieren, z.B. um Daten von/auf WSDL-Operationen zu mappen und Programmcode in Java oder C# zu generieren, der den Webservice implementiert. Die Ergebnisse können anschließend im Fenster "Ausgabe" (d.h. im Built-In-Ausführungsprozessor) angezeigt werden. Nun müssen Sie den generierten Code nur noch kompilieren und den Webservice auf Ihrem Webserver bereitstellen.

MapForce unterstützt WSDL-Projekte, die aus einzelnen Mappings bestehen, von denen jedes eine Webservice-Operation darstellt. Wenn Sie Code für das gesamte WSDL-Projekt generieren, wird ein vollständiger Webservice-Server erzeugt. Es kann jedoch zu Testzwecken auch Code für einzelne Operationen/Mappings generiert werden.

Beachten Sie bitte, dass in diesem Dokument nicht auf die Installation oder Konfiguration der erforderlichen Webservice-Software eingegangen wird. Lesen Sie zu diesem Thema bitte die Dokumentation zur jeweiligen Software oder bitten Sie Ihre IT-Abteilung, die Software für Sie zu installieren.

Für das Generieren und Bereitstellen von z.B. Java-Webservices benötigte Software:

- MapForce Enterprise Edition
- Java 2 Software Development Kit (1.5 oder höher): <http://java.sun.com/j2se/>
- Apache Tomcat: <http://tomcat.apache.org>
- Apache Axis2: <http://ws.apache.org/axis2/>, ein SOAP Framework, das innerhalb von Tomcat ausgeführt wird
- Apache ANT: <http://ant.apache.org/>

Für die Generierung von C# Webservices benötigte Software:

- MapForce Enterprise Edition
  - Microsoft Visual Studio 2005 oder höher
  - Microsoft Internet Information Services (IIS) Version 5.0, oder höher.

Die MapForce-Projektdaten und alle anderen Dateien, die in diesem Abschnitt verwendet werden, finden Sie im Ordner **...\\MapForceExamples\\Tutorial**.



## 24.1 WSDL - Info - unterstützte Protokolle

Die WSDL-Datei zur Erzeugung eines Webservice beschreibt die Schnittstelle zum Webservice vollständig. Die WSDL-Datei, auf der der Webservice basiert, muss außerhalb von MapForce erstellt werden; Sie können dafür z.B. XMLSpy verwenden.

### PortType (WSDL 1.1)

Ein <portType> Element definiert eine Webservice-Schnittstelle, d.h. es:

- definiert die **Operationen**, die ausgeführt werden können
- die **Messages**, die in den einzelnen Operationen als Inputs und Outputs verwendet werden.

### Types: (WSDL 1.1)

Das <types> Element definiert die Datentypen, die vom Webservice verwendet werden. MapForce unterstützt XML-Schemas in WSDL-Dateien, da dies das gebräuchlichste Typsystem für WSDL-Dateien ist. In MapForce werden diese Elemente (Datentypen) als Datenelemente in einer (Message)-Komponente angezeigt, sodass Sie diese direkt auf andere Datenelemente/Konstrukte mappen können.

### Messages: (WSDL 1.1)

Das <message> Element definiert die **Parts** (Teile) jeder Message sowie die **Datenelemente** der Input- und Output-Parameter einer Operation. Dies sind die Messages, die zwischen Client und Server ausgetauscht werden. Es gibt drei Arten von Messages: Input, Output und Fault.

### Operationen: (WSDL 1.1)

In Operationen werden Messages als Input- und Output-Parameter verwendet. Eine Operation kann die folgenden Messages enthalten:

- eine Input Message
- null oder mehr Output Messages
- null oder mehr Fault Messages

Bitte beachten Sie:

- **Input** Messages können nur als **Quellkomponenten** verwendet werden
- **Output-** und **Fault** Messages können nur als **Zielkomponenten** verwendet werden

In MapForce ist jede **Operation** ein separates **Mapping**-Dokument mit ihrer eigenen \*.mfd-Datei. Die Sammlung von Operationen wird in einer MapForce WSDL-Projektdatei gruppiert, welche den kompletten Webservice beschreibt.

**WSDL 2.0** unterscheidet sich beträchtlich von WSDL 1.1. Die wichtigsten Unterschiede sind:

- PortTypes wurden in Interfaces umbenannt.
- Messages und parts werden nun anhand des XML-Schematyps im Element types definiert
- Ports wurden in Endpoints umbenannt.
- WSDL 2.0 Operation Inputs und Outputs werden vom XML-Schema definiert.

Im Dialogfeld "Komponenteneinstellungen" einer WSDL-Komponente (in MapForce) wird sowohl für WSDL 1.1 Ports als auch für WSDL 2.0 Endpoints "Endpoint" angezeigt.

WSDL-Unterstützung: Version 1.1 und 2.0, W3C-Anmerkung von  
<http://www.w3.org/TR/wsdl>

|                                                                                                                                                                            |                                                                                                                                                                                                                                                                                           |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                                                                                            | version 2.0, W3C Recommendation von<br><a href="http://www.w3.org/TR/wsdl20/">http://www.w3.org/TR/wsdl20/</a>                                                                                                                                                                            |
| WSDL Type-System:                                                                                                                                                          | XML Schema 2001                                                                                                                                                                                                                                                                           |
| SOAP-Unterstützung:                                                                                                                                                        | Version 1.1:<br><a href="http://www.w3.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3.org/TR/2000/NOTE-SOAP-20000508/</a><br>Version 1.2: <a href="http://www.w3.org/TR/soap12-part0/">http://www.w3.org/TR/soap12-part0/</a>                                                             |
| Protokolle:                                                                                                                                                                | SOAP über HTTP (HTTP POST, HTTP GET Protokolle werden nicht unterstützt).                                                                                                                                                                                                                 |
| C#                                                                                                                                                                         | Die SOAPAction muss für jede Operation in C# anders sein                                                                                                                                                                                                                                  |
| Bindings:                                                                                                                                                                  | Derzeit werden mehrere Operationen mit demselben Namen nicht unterstützt (WSDL 2.5).                                                                                                                                                                                                      |
| Style/Verwendung:                                                                                                                                                          | <ul style="list-style-type: none"> <li>• document/literal: unterstützt.</li> <li>• RPC/literal: in C# unterstützt</li> <li>• RPC/encoded: eingeschränkte Unterstützung</li> <li>• Derzeit wird ein Stil/eine Verwendung pro Webservice (Java) oder Operation (C#) unterstützt.</li> </ul> |
| SOAP Headers:                                                                                                                                                              | Abhängig vom verwendeten Betriebssystem.                                                                                                                                                                                                                                                  |
| SOAP encodingStyle:                                                                                                                                                        | Wenn use="encoded", wird als Kodierungsstil für den kompletten soap:Body "http://schemas.xmlsoap.org/soap/encoding/" angenommen. Andere Kodierungsstile werden nicht unterstützt.                                                                                                         |
| SOAP:                                                                                                                                                                      | encodingStyle-Attribut wird in Messages ignoriert (SOAP 4.1.1).                                                                                                                                                                                                                           |
| SOAP encoding:                                                                                                                                                             | <ul style="list-style-type: none"> <li>• href auf externe Ressourcen werden derzeit nicht unterstützt (SOAP 5.4.1).</li> <li>• Referenzen werden nur auf unabhängige Elemente unterstützt</li> </ul>                                                                                      |
| SOAP-ENC:Array:                                                                                                                                                            | Linearer Zugriff unterstützt; partial arrays, sparse arrays werden derzeit nicht unterstützt.                                                                                                                                                                                             |
| Benutzerdefinierte SOAP-Verbesserungen:                                                                                                                                    | nicht unterstützt.                                                                                                                                                                                                                                                                        |
| Default, oder fixed values in Schemas:                                                                                                                                     | nicht unterstützt.                                                                                                                                                                                                                                                                        |
| Validierung von nicht-SOAP Messages                                                                                                                                        | werden nicht validiert, sondern an das zugrundeliegende Framework übergeben.                                                                                                                                                                                                              |
| Namespaces                                                                                                                                                                 | Nicht-Namespace-Einträge sind ungültige WSDL und werden daher nicht unterstützt (WSDL und XML 1.0)                                                                                                                                                                                        |
| Bitte beachten Sie: (WSDL 1.1)                                                                                                                                             |                                                                                                                                                                                                                                                                                           |
| Bei Verwendung der Kombination Dokument / Literal in MapForce, muss das <b>Message / Part Element</b> ein globales Element im Gegensatz zu einem <b>Typ</b> referenzieren. |                                                                                                                                                                                                                                                                                           |

Das "element"-Attribut bezieht sich auf ein in einem Schema definiertes globales Element (ns2:Vendor):

```
<message name="processRequest">
 <part name="inputData" element="ns2:Vendor"/>
</message>
```

Während das folgende einen Typ im Schema referenziert.:

```
<message name="processRequest">
 <part name="inputData" type="ns2:VendorType"/>
</message>
```

## 24.2 Erstellen von Webservice-Projekten anhand von WSDL-Dateien

### Ziel des Webservice-Projekts:

Erstellung eines Webservice, mit Hilfe dessen ein Benutzer folgendes tun kann:

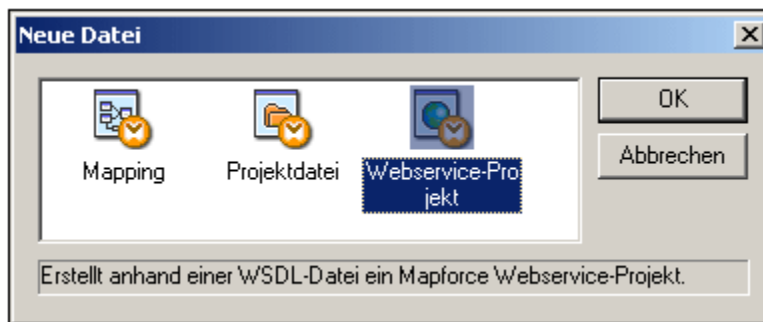
- Suchen nach einer bestimmten Gruppe von Personen in einer MS Access Datenbank auf einem Server mittels SOAP Request. (Die Abfrage wird in Laufzeit am Client eingegeben/definiert und anschließend an den Server gesendet.)
- Abrufen der Datenbank-Datensätze als SOAP Response in Form eines XML-Dokuments, das an den Client zurückgesendet wird und alle Personen enthält, die die Abfragekriterien erfüllen.

Bitte beachten Sie:

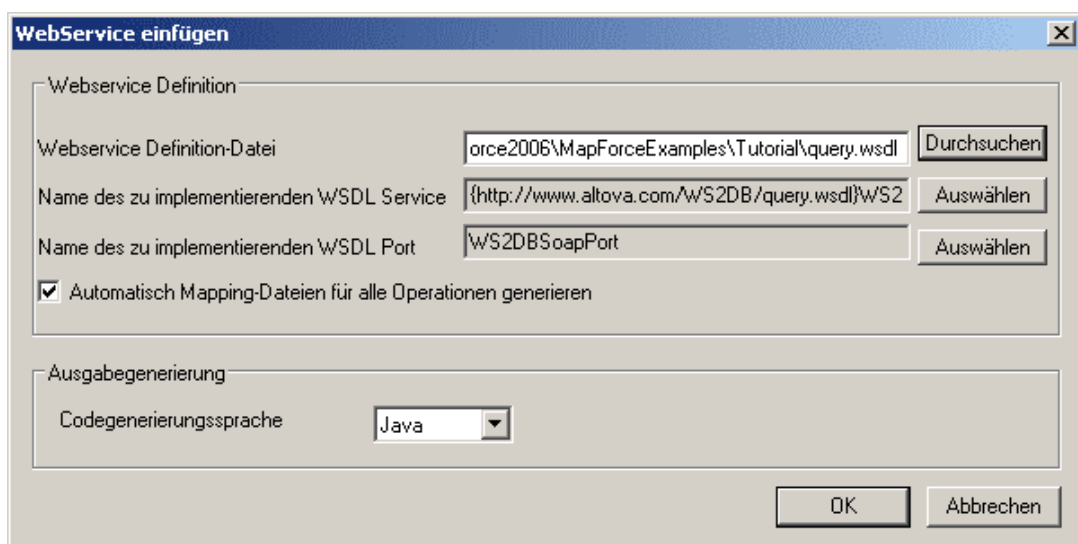
Das Mapping, mit dem der Webservice generiert wird, hängt nicht von der Ziel-Programmiersprache ab, d.h. es ist identisch, egal ob Java- oder C#-Webservices generiert werden. Der Unterschied entsteht erst beim Kompilieren und Bereitstellen des Webservice auf dem Webserver.

### Erstellen eines Webservice-Projekts:

1. Wählen Sie die Menüoption **Datei | Neu**.
2. Klicken Sie auf das Symbol "Webservice-Projekt" und bestätigen Sie mit OK.



3. Füllen Sie das Dialogfeld "Webservice einfügen" aus.

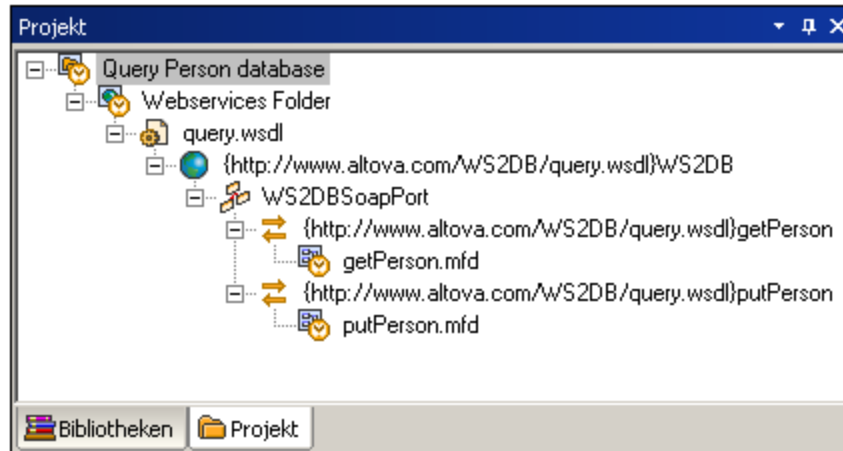


4. Wählen Sie im Bereich Webservice-Definition die WSDL-Datei **query.wsdl** aus. **Query.wsdl** liegt im Verzeichnis **...MapForceExamples\Tutorial\**. Bei Auswahl der WSDL-Datei werden die restlichen Felder automatisch ausgefüllt. Wenn Ihre

WSDL-Datei mehrere mögliche Optionen enthält, wird ein Dialogfeld angezeigt, in dem Sie den gewünschten WSDL-Service oder Port auswählen können.

5. Klicken Sie auf OK.
6. Geben Sie den Namen des WSDL-Projekts im Dialogfeld "Projekt speichern unter..." ein.

Klicken Sie auf "Speichern", um die Einstellungen zu bestätigen und die WSDL-Projektdatei zu erstellen.



Auf dem Projektregister sehen Sie das Projekt und den WSDL-Namen sowie die einzelnen in der WSDL-Datei definierten Operationen. Die beiden Operationen sind **getPerson** und **putPerson**.

#### Erstellen von Webservice Mappings:

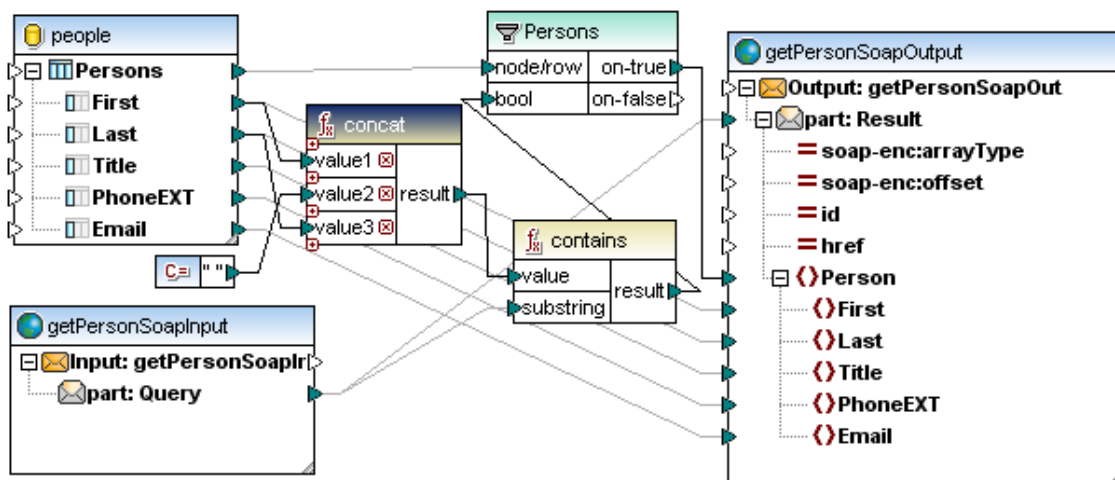
1. Doppelklicken Sie auf dem Projektregister auf die Operation **WS2DB\_getPerson**. Es wird ein neues Mapping "getPerson.mfd" erstellt, das die zwei WSDL-Komponenten enthält. Die **getPersonSoapIn**-Komponente enthält das Datenelement query, mit Hilfe dessen die Datenbank über den Webservice abgefragt werden soll. Die **getPersonSoapOut**-Komponente enthält die Datenelemente "Person", die in der WSDL-Datei definiert sind.



#### Definieren des Mappings für die Datenbankabfrage:

1. Wählen Sie **Einfügen | Datenbank** und fügen Sie die Datenbank **people.mdb** (in Ordner ...\\Tutorial) mit Hilfe des Assistenten ein. Die Datenelemente in der Datenbank entsprechen denjenigen in der Komponente **getPersonSoapOut**.
2. Mappen Sie die Person-Datenelemente auf die entsprechenden Datenelemente in der

- Komponente `getPersonSoapOut`.
- Verketten Sie die Vor- und Nachnamen (First und Last) der Personen mit Hilfe der `concat`-Funktion.
  - Fügen Sie eine **"contains"**-Funktion ein und verbinden Sie das **result**-Element der `concat`-Funktion mit dem **value**-Parameter.
  - Klicken Sie auf das **part: Query**-Element in der `getPersonSoapIn`-Komponente und verbinden Sie es mit dem **substring**-Element der **"contains"**-Funktion.
  - Klicken Sie erneut auf das **part: Query**-Element und verbinden Sie es mit dem Datenelement **part: Result** der `getPersonSoapOutput`-Komponente.
- Das Query-Datenelement/Element der `getPersonSoapIn`-Komponente ist der Platzhalter für die Abfrage, d.h. hier wird im SOAP Client der Abfragestring eingegeben, sobald der Code generiert, kompiliert und auf dem Webserver bereitgestellt wurde.
- Fügen Sie eine filter-Komponente ein und stellen Sie das Mapping fertig, wie im Diagramm unten gezeigt.



Sie können nun Code generieren, um einen Webservice zu erstellen. Webservices können derzeit für [Java](#) oder [C#](#) generiert werden. Im folgenden Abschnitt wird die Generierung und Kompilierung des Codes sowie die Bereitstellung der jeweiligen Webservices für Java und C# beschrieben.

## 24.3 Generieren von Java Webservices mit MapForce

MapForce generiert den gesamten Code und alle Scripts, die zur Erstellung eines Webservice benötigt werden. Der einzige Unterschied zur normalen Codegenerierung ist, dass der generierte Code auf dem Axis2 (Tomcat)-Server bereitgestellt werden muss.

Beachten Sie Folgendes, wenn der Webservice auf einem remote Rechner ausgeführt wird:

- Der Benutzer benötigt remote Administratorrechte
- Das Axis2 Framework muss (zum Teil) auf dem lokalen Computer installiert werden
- Wenn Webservices auf einem lokalen Rechner bereitgestellt werden, d.h. wenn der Server der lokale Computer ist, sind diese Punkte nicht relevant.
- Axis2 unterstützt sowohl SOAP1.1 als auch SOAP1.2.

### Generieren von Java-Code:

Nachdem Sie die im vorherigen Abschnitt verwendete Projektdatei **Query Person database.mfp** erstellt bzw. geöffnet haben:

1. Klicken Sie mit der rechten Maustaste im Projektfenster auf den Projektnamen und wählen Sie **Code generieren in | Java**.
2. Erstellen Sie den generierten Java-Code (z.B. mit Apache Ant), um die \*.aar Datei für die Bereitstellung auf Axis2 zu erzeugen.

Die folgenden Ordner und Dateien werden automatisch im Zielverzeichnis erstellt:

- ein **com**-Verzeichnis mit den Unterverzeichnissen **altova** und **MapForce**.

### Online-Bereitstellung von Axis2

Verwenden Sie entweder den Upload-Service auf der Administrator-Seite, um den von MapForce als \*.aar-Datei generierten Webservice hinaufzuladen.

oder

führen Sie den Upload manuell durch:

Wenn Ihre Tomcat-Version im Ordner:

C:\Program Files\Apache Software Foundation\Tomcat 5.5,

installiert wurde, können Sie die Datei manuell in den Ordner

C:\Program Files\Apache Software Foundation\Tomcat 5.5\webapps\axis2\WEB-INF\services

kopieren

### Webservice online entfernen

Löschen Sie die Datei \*.aar aus dem Ordner <tomcat path>\webapps\axis2\WEB-INF\services.

### Axis2-Einschränkungen

Die Axis2-Unterstützung für RPC/encode ist beschränkt. MapForce kann jedoch RPC/encode WebServices (sowohl SOAP 1.1 als auch soap 1.2) generieren, mit der Einschränkung, dass die Original-WSDL-Datei nicht vom Webserver abgerufen wird.

D.h., dass z.B. <http://127.0.0.1/axis2/services/WS2DB?wsdl> keine verwendbare WSDL-Datei zurückgeben würde.

Bei **document literal** WebServices stellt die obige URL eine brauchbare, korrekte WSDL-Datei bereit. Sie unterscheidet sich jedoch vom Original: Kommentare werden entfernt, Namespaces werden geändert. Der Webservice hat jedoch immer noch dieselbe Semantik wie die Original-WSDL-Datei, mit der der Service erstellt wurde.

Zwar unterstützt Axis2 RPC/encoded nicht, doch kann es WSDL von online gestelltem Java-Code (kompiliertem Code) generieren. Daher kann MapForce-generierter Code RPC/encoded Messages verarbeiten; Axis2 wird nur für die Übertragung verwendet.

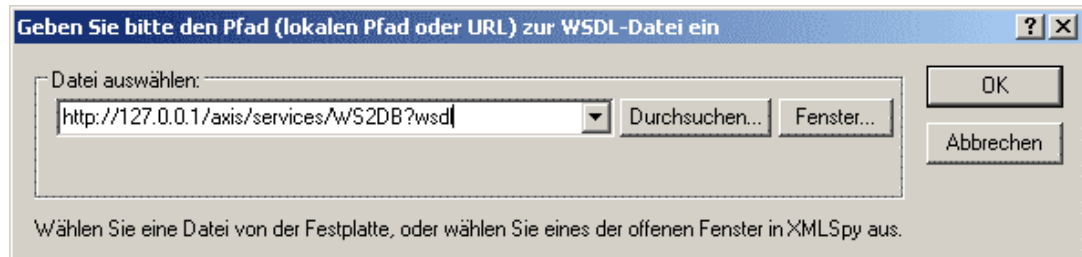


### 24.3.1 Verwendung des Webservice - getPerson-Operation

#### Verwendung des Webservice:

Mit Hilfe eines WSDL Client z.B. XMLSpy können Sie einen SOAP Request erstellen und senden.

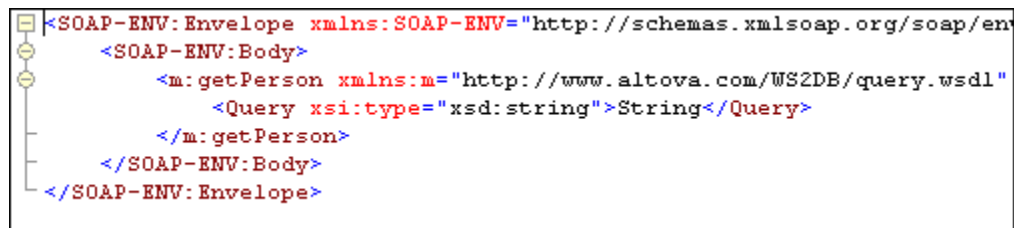
1. Wählen Sie **SOAP | SOAP Request erzeugen**.
2. Geben Sie den Pfad zur WSDL-Datei auf dem Server ein, z.B.  
**http://127.0.0.1/axis2/services/WS2DB?wsdl** und bestätigen Sie mit OK.



3. Wählen Sie den Namen der gewünschten SOAP-Operation aus, in diesem Fall "**getPerson( string Query)**", und klicken Sie auf OK, um das SOAP Request-Dokument zu erzeugen.

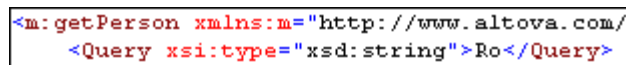


Die Zeile mit `<Query xsi:type="xsd:string">String</Query>` ist der Platzhalter für die Abfrage bzw. der String, nach dem Sie in der Datenbank suchen möchten.



Wir möchten vom Webservice alle Person-Datensätze erhalten, die entweder im Vor- oder Nachnamen (First oder Last) die Zeichenfolge "Ro" enthalten.

4. Bearbeiten Sie den String-Text und geben Sie z.B. **Ro** ein.



5. Wählen Sie **SOAP | SOAP Request an Server senden**, um den Request an den Server zu senden.



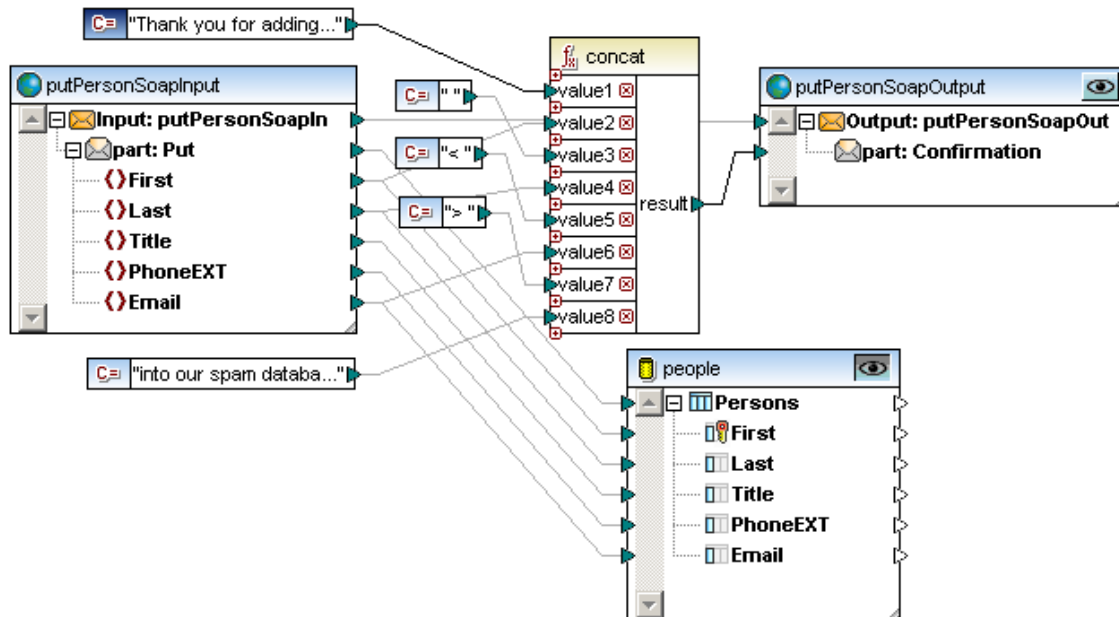
```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
 <soapenv:Body>
 <m:getPersonResponse xmlns:m="http://www.altova.com/WS2DB/">
 <Result soapenc:arrayType="n0:Person[]" xsi:type="soapenc:Array">
 <Person xsi:type="n0:Person">
 <First xsi:type="xsd:string">Martin</First>
 <Last xsi:type="xsd:string">Rope</Last>
 <Title xsi:type="xsd:string">Mr.</Title>
 <PhoneEXT xsi:type="xsd:string">780</PhoneEXT>
 <Email xsi:type="xsd:string">mr@strings.com</Email>
 </Person>
 <Person xsi:type="n0:Person">
 <First xsi:type="xsd:string">Ronald</First>
 <Last xsi:type="xsd:string">Superstring</Last>
 <Title xsi:type="xsd:string">Dr.</Title>
 <PhoneEXT xsi:type="xsd:string">444</PhoneEXT>
 <Email xsi:type="xsd:string">ros@strings.com</Email>
 </Person>
 <Person xsi:type="n0:Person">
 <First xsi:type="xsd:string">Robert</First>
 <Last xsi:type="xsd:string">Darkmatter</Last>
 <Title xsi:type="xsd:string">Mathematician</Title>
 <PhoneEXT xsi:type="xsd:string">299</PhoneEXT>
 <Email xsi:type="xsd:string">rodark@strings.com</Email>
 </Person>
 <Person xsi:type="n0:Person">
 <First xsi:type="xsd:string">Roger</First>
 <Last xsi:type="xsd:string">Gravity</Last>
 <Title xsi:type="xsd:string">Crisis manager</Title>
 <PhoneEXT xsi:type="xsd:string">112</PhoneEXT>
 <Email xsi:type="xsd:string">rog@strings.com</Email>
 </Person>
 </Result>
 </m:getPersonResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Das SOAP Response-Dokument wird vom Webservice zurückgegeben und enthält vier Personen aus der Access-Datenbank.

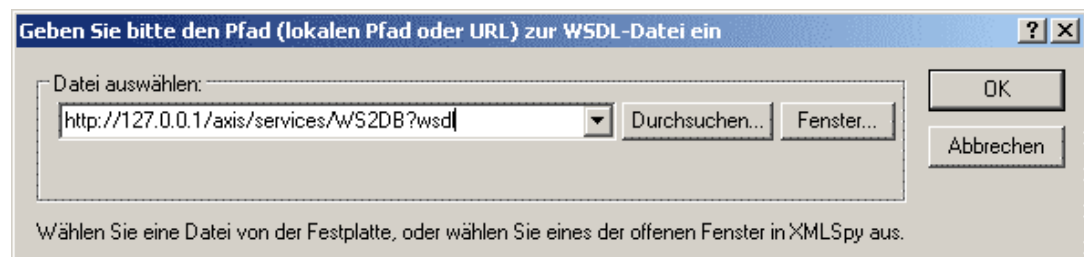
### 24.3.2 Verwendung des Webservice - putPerson-Operation

- Doppelklicken Sie im Projektfenster auf das **putPerson**-Mapping-Symbol (unterhalb des Ordners Mappings), um das Mapping anzuzeigen.

Daraufhin wird die Datei **putPerson.mfd** auf einem zusätzlichen Register geöffnet.



- Mit Hilfe eines WSDL Client wie z.B. XMLSpy erstellen und senden Sie einen SOAP Request.
- Wählen Sie **SOAP | SOAP Request erzeugen**.
- Geben Sie den Pfad zur WSDL-Datei auf dem Server ein, z.B. **http://127.0.0.1/axis2/services/WS2DB?wsdl** und klicken Sie auf OK.



- Wählen Sie den Namen der gewünschten SOAP-Operation aus, in diesem Fall **"putPerson (Person Put)"**, und klicken Sie auf OK, um das SOAP Request-Dokument zu erstellen. Das SOAP Request-Dokument liefert die Platzhalter für die Person-Elemente, die der Benutzer ausfüllen muss.

```

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
 <m:putPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl">
 <Put xsi:type="m0:Person">
 <First xsi:type="xsd:string">String</First>
 <Last xsi:type="xsd:string">String</Last>
 <Title xsi:type="xsd:string">String</Title>
 <PhoneEXT xsi:type="xsd:string">String</PhoneEXT>
 <Email xsi:type="xsd:string">String</Email>
 </Put>
 </m:putPerson>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

5. Ersetzen Sie den Platzhaltertext (String) durch die Personendaten.

```

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
 <m:putPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl">
 <Put xsi:type="m0:Person">
 <First xsi:type="xsd:string">Fred</First>
 <Last xsi:type="xsd:string">Flagellator</Last>
 <Title xsi:type="xsd:string">Sir</Title>
 <PhoneEXT xsi:type="xsd:string">777</PhoneEXT>
 <Email xsi:type="xsd:string">ff@home.com</Email>
 </Put>
 </m:putPerson>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

6. Wählen Sie **SOAP | SOAP Request an Server senden**, um den Request an den Server zu senden.

Mit dem Befehl "SOAP Request an Server senden" wird das editierte SOAP Request-Dokument an den Server gesendet.

Die neuen Personendaten werden anschließend zur Datenbank hinzugefügt und ein SOAP Response-Dokument wird als Bestätigung zurückgesendet. Im Mapping wird der Inhalt der Bestätigungsmeldung definiert, z.B. Vielen Dank, dass Sie XYZ zu unserer Spam-Datenbank hinzugefügt haben.

```

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
 <m:putPersonResponse xmlns:m0="http://www.altova.com/WS2DB.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:m="http://www.altova.com/WS2DB/query">
 <Confirmation xsi:type="xsd:string">Thank you for adding Fred Flagellator <ff@home.com> into our spam database</Confirmation>
 </m:putPersonResponse>
 </soapenv:Body>
</soapenv:Envelope>

```

## 24.4 Generieren von C# Webservices mit MapForce

Die Datei `query.wsdl` im Ordner `Tutorial` hat einen Abschnitt, der auskommentiert ist, nämlich den Abschnitt `<Service name="WS2DB">` am Ende der Datei. Aktivieren Sie diesen auskommentierten Abschnitt wieder und kommentieren Sie den vorhergehenden `<Service...>` aus, bevor Sie mit diesem Beispiel beginnen. In diesem Beispiel wird davon ausgegangen, dass sich der Webserver auf dem lokalen Rechner befindet.

Bitte beachten Sie:

Die URI-Schemas für Axis2 (Java) und IIS (C#) sind unterschiedlich. Die WSDL-Datei muss dafür angepasst werden. Die Unterschiede befinden sich im Element `<service>`, das Sie normalerweise am Ende der WSDL-Datei finden.

### WSDL-Datei für Axis2:

```
<service name="PersonWS">
 <port name="PersonWSQuerySoapPort" binding="tns:PersonWSSoapBinding">
 <soap:address location="http://localhost:8080/axis2/services/PersonWS"/>
 </port>
</service>
```

### WSDL-Datei für IIS:

```
<service name="PersonWS">
 <port name="PersonWSQuerySoapPort" binding="tns:PersonWSSoapBinding">
 <soap:address location="http://localhost/services/PersonWS.asmx"/>
 </port>
</service>
```

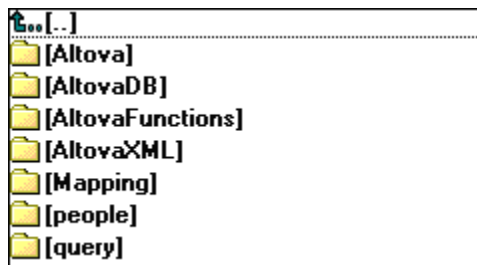
Die WSDL-Beispieldatei enthält beide Varianten. Eine davon ist auskommentiert. Außerdem enthält die WSDL-Beispieldatei den Parameter `"soapAction"`, der für IIS benötigt wird, von Axis2 aber ignoriert wird.

### Generieren von C# Code:

Nachdem Sie die Projektdatei `Query Person database.mfp` aus dem vorherigen Abschnitt erstellt bzw. geöffnet haben:

1. Wählen Sie **Datei | Code generieren in | C#**.
2. Wählen Sie das Ausgabeverzeichnis, in diesem Fall `cs-dev` oder `C#-dev` und klicken Sie auf OK, um die Codegenerierung zu starten.  
Nach erfolgreicher Codegenerierung erscheint die Meldung "C#-Code-Generierung abgeschlossen". Andernfalls wird eine Meldung mit den/dem jeweiligen Fehler(n) angezeigt.

Im Zielverzeichnis werden mehrere Ordner erstellt, die den C#-Code, die Projektdateien, die Dateien mit der Lösung sowie die Visual Basic Skripts enthalten.



Sie können das generierte Projekt in Visual Studio 2005/2008/2010 kompilieren.

### Kompilieren des Webservice:

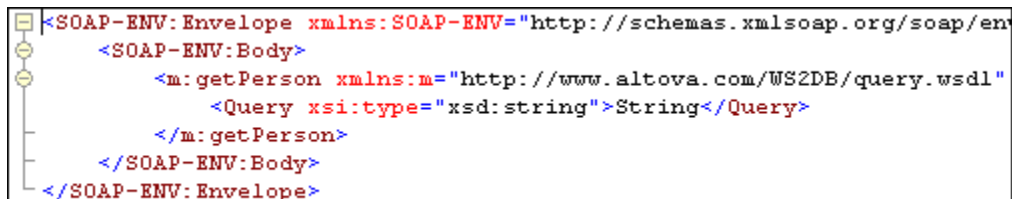
Bevor Sie die Datei mit der Lösung in Visual Studio öffnen, müssen Sie das Visual Basic Skript

**virtidir.vbs** einmal ausführen. Das Skript erstellt auf dem Server ein virtuelles Verzeichnis, das auf den Ordner verweist, in dem der Code generiert wurde. Wenn der Server nicht auf dem lokalen Rechner installiert ist, muss der Code auf den Webserver kopiert werden und das virtuelle Verzeichnis muss dort erstellt werden.

1. Navigieren Sie zum Unterverzeichnis **Mapping** und führen Sie die Skriptdatei **virtidir.vbs** aus.  
Dadurch wird ein virtuelles Verzeichnis für IIS erstellt, das den generierten Code referenziert.
2. Öffnen Sie die Datei mit der Mapping-Lösung **..\Mapping\Mapping\_webservice.sln**.
3. Wählen Sie die Menüoption **Build | Build Solution**, um das Webservice-Projekt zu kompilieren.
4. Wählen Sie die Menüoption **Debug | Run**, um die Applikation zu starten.  
Wenn sich der Webservice auf einem remote Computer befindet, muss der kompilierte Code in den entsprechenden Ordner auf dem Server kopiert werden.

#### Verwendung des Webservice:

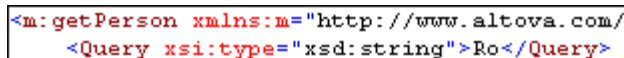
1. Verwenden Sie einen WSDL Client, z.B. XMLSpy, um einen SOAP Request zu erstellen und zu senden.
2. Wählen Sie die Menüoption **SOAP | SOAP Request erzeugen**.
3. Geben Sie den Pfad zur WSDL-Datei auf dem IIS Server ein und bestätigen Sie mit OK.
4. Wählen Sie den Namen der gewünschten SOAP Operation aus, in diesem Fall "**getPerson( string Query)**", und klicken Sie auf OK, um das SOAP Request-Dokument zu erzeugen.  
Die Zeile mit dem Codefragment `<Query xsi:type="xsd:string">String</Query>` ist der Platzhalter für die Abfrage bzw. den String, nach dem Sie in der Datenbank suchen möchten.



```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
 <m:getPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl">
 <Query xsi:type="xsd:string">String</Query>
 </m:getPerson>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Wir möchten vom Webservice alle Personen-Datensätze erhalten, die entweder im Vor- oder Nachnamen ("First" oder "Last") den String "Ro" enthalten.

4. Bearbeiten Sie den Sting-Text und geben Sie z.B: **Ro** ein.



```
<m:getPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl">
 <Query xsi:type="xsd:string">Ro</Query>
```

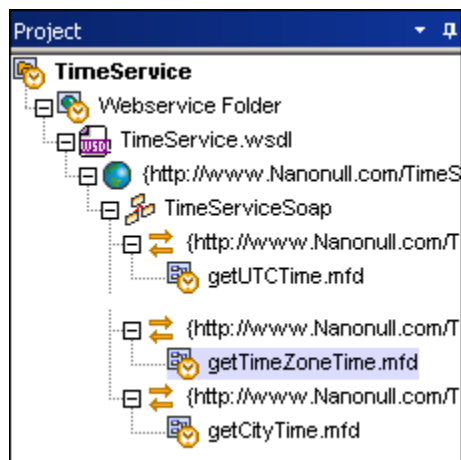
5. Wählen Sie **SOAP | SOAP Request an Server senden**, um den Request an den Server zu senden.  
Das vom Webservice zurückgegebene SOAP Response-Dokument enthält vier Personen-Datensätze aus der Access-Datenbank.

## 24.5 Webservice Fehler

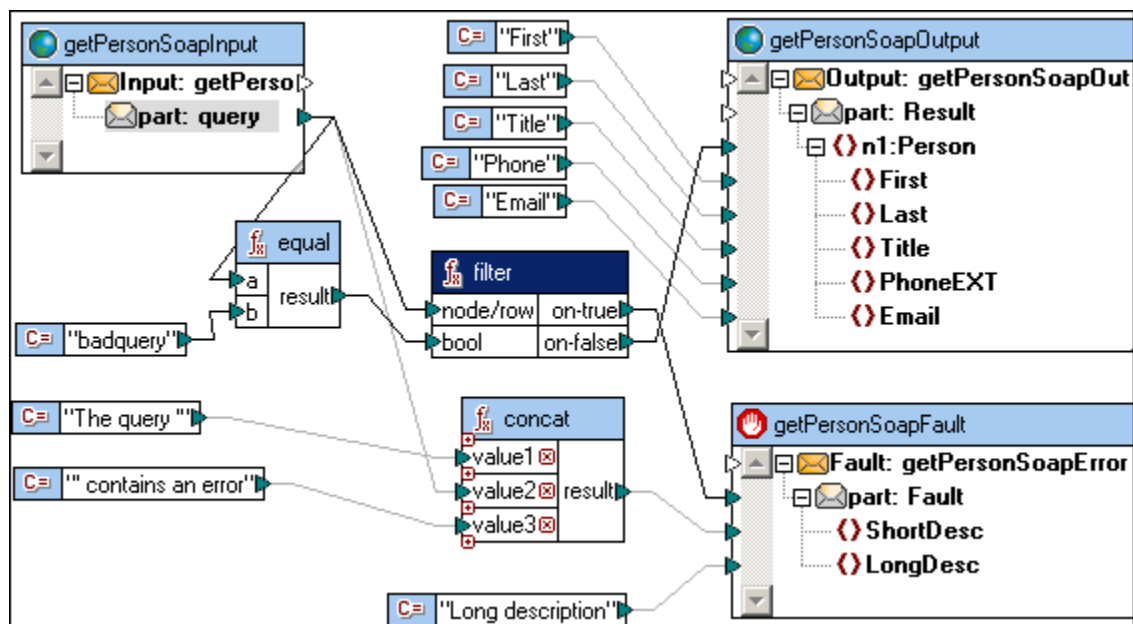
MapForce unterstützt die Definition von WSDL-Fehlern. Eine WSDL-Datei kann ein Fehler-Element für eine Operation enthalten sowie ein Meldungsattribut, das die Fehlermeldung enthält. Ein Fehler-Element muss in der WSDL-Datei vorhanden sein, damit Sie in ein Mapping eine Fehler-Komponente einfügen können.

Sie können die Bedingung definieren, durch die ein Fehler ausgelöst wird. Wenn die Bedingung zutrifft, wird im Meldungsfenster eine benutzerdefinierte Meldung angezeigt und der Mapping-Vorgang wird gestoppt.

- Doppelklicken Sie im Projektfenster auf den Eintrag **getTimeZoneTime.mfd file**. Ein Beispiel für eine Fehlerkomponente finden Sie im Projekt **Timeservice.mfp** aus dem Ordner ...MapForceExamples\TimeService.



Daraufhin wird das Mapping geöffnet, in dem Sie sehen, wie Webservice-Fehler verwendet werden. (In der Abbildung sehen Sie WSDL 1.1)



Im obigen Beispiel sehen Sie, wie Ausnahmeereignisse in Mappings definiert werden. Das

Ausnahmeereignis sollte ausgelöst werden, wenn n0:Timezone nicht gleich "UTC" ist.

- Die **equal**-Komponente überprüft, ob timezone gleich UTC ist, das bool-Ergebnis wird an die filter-Komponente weitergegeben.
- Wenn die Bedingung false ist, d.h. ein anderer Wert als UTC ist, aktiviert der **on-false**-Parameter der filter-Komponente das Ausnahmeereignis Fault:errorSoapOut und der Mapping-Vorgang wird gestoppt. (Beachten Sie, dass Sie das Ausnahmeereignis bei Bedarf auch mit dem on-true-Parameter verbinden können.)
- Die SoapFault-Message liefert zwei Fehlertext-Gruppen.

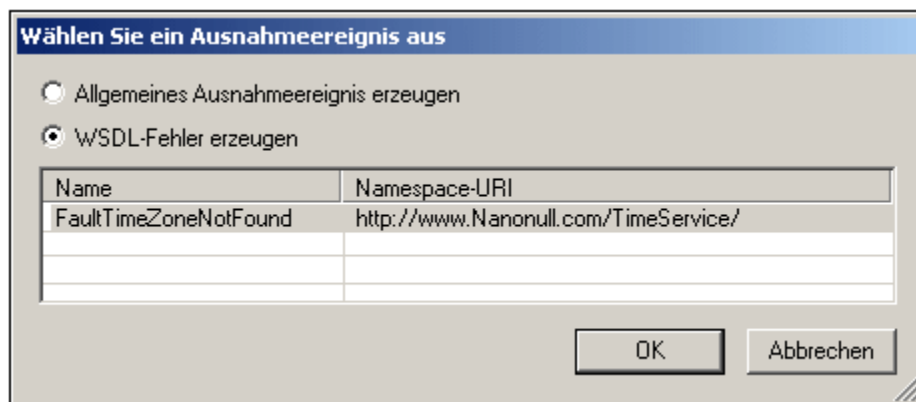
Bitte beachten Sie:

Es ist äußerst wichtig, wo Sie in diesem Beispiel die filter-Komponente platzieren:

- Es müssen **beide Parameter** der filter-Komponente, sowohl on-true als auch on-false, gemappt werden! Einer muss auf die fault-Komponente gemappt werden und der andere auf die Zielkomponente, an die die gefilterten Daten der Quellkomponente übergeben werden. Andernfalls wird die fault-Komponente nie ausgelöst.
- Die **Ausnahmeereigniskomponente** und die **Zielkomponente** müssen direkt mit der **filter**-Komponente verbunden werden. Zwischen die filter-Komponente und die Ausnahmeereignis- bzw. die Zielkomponente dürfen keine Funktionen oder andere Komponenten platziert werden.

#### So fügen Sie eine WSDL-Fehlerkomponente ein:

1. Wählen Sie die Menüoption **Einfügen | Ausnahme** oder klicken Sie in der Symbolleiste auf das Symbol "Ausnahmeereignis einfügen" .
2. Klicken Sie auf das Optionsfeld "XSDL-Fehler erstellen", wählen Sie anschließend den Namen des Fehlers in der Liste aus und klicken Sie auf OK , um die Fehlerkomponente einzufügen.



| Name                  | Namespace-URI                        |
|-----------------------|--------------------------------------|
| FaultTimeZoneNotFound | http://www.Nanonull.com/TimeService/ |
|                       |                                      |
|                       |                                      |



# Kapitel 25

---

## Aufrufen von Webservices

## 25 Aufrufen von Webservices

MapForce unterstützt das direkte Aufrufen von Webservice-Funktionen von einem Mapping aus. Dies bedeutet, dass Sie eine Webservice-Funktion in ein Mapping einfügen können, Input- und Output-Komponenten damit verbinden können und das Ergebnis des Aufrufs der Webservice-Funktion sofort im Ausgabefenster sehen.

Beachten Sie, dass der Webservice, wie im Abschnitt [Aufruf des Webservices getPerson](#) gezeigt, sowohl mit einer Input- als auch einer Output-Komponente verbunden sein muss. Wenn eine der Komponenten fehlt, schlägt die Validierung des Mappings fehl.

Webservice-Funktionen werden nur in generiertem Java- und C#-Code unterstützt, eine Vorschau davon kann aber im Built-In-Ausführungsprozessor angezeigt werden (Schaltfläche "BUILTIN").

In diesem Abschnitt erfahren Sie Folgendes:

- Wie man an einem Timeservice mit Hilfe einer Konstante eine Abfrage durchführt
- Wie man eine Abfrage an einem Webservice durchführt, welcher die Ergebnisse aus einer Datenbank liefert

Webservice-Aufrufe in MapForce unterstützen derzeit die folgenden Protokolle:

### SOAP 1.1 und 1.2

- RPC / encoded und Dokument / Literal
- Behandlung von Referenzen (RPC/encoded)
- Fehlerbehandlung: Auffinden von Wsdl-defined- und Nicht-Wsdl-Fehlern

Wsdl-Fehler können auf eine Ausnahmeereigniskomponente gemappt werden; stoppen die Ausführung

Nicht-Wsdl-Fehler werden am Bildschirm angezeigt; stoppen die Ausführung

### Nicht-SOAP

- HTTP GET: url-encoded
- HTTP POST: url-encoded und Text/xml

### Allgemeine Features

- Authentifizierung (nur einfache Authentifizierung)
- Vom Server zurückgegebene Fehler werden behandelt (Fehler bei denen es sich nicht um "Faults" handelt, also falsche URL, Timeout, usw.)

In diesem Abschnitt werden die folgenden Dateien verwendet, die im MapForce Installationsordner zur Verfügung stehen:

- **TimeService.wsdl** Webservice im Ordner .....**MapForceExamples\Timeservice**
- **TimeService2.0.wsdl** Webservice im Ordner ...**MapForceExamples\TimeserviceWsdI2**
- **Query.wsdl** Webservice im Ordner ...**MapForceExamples\Tutorial\**

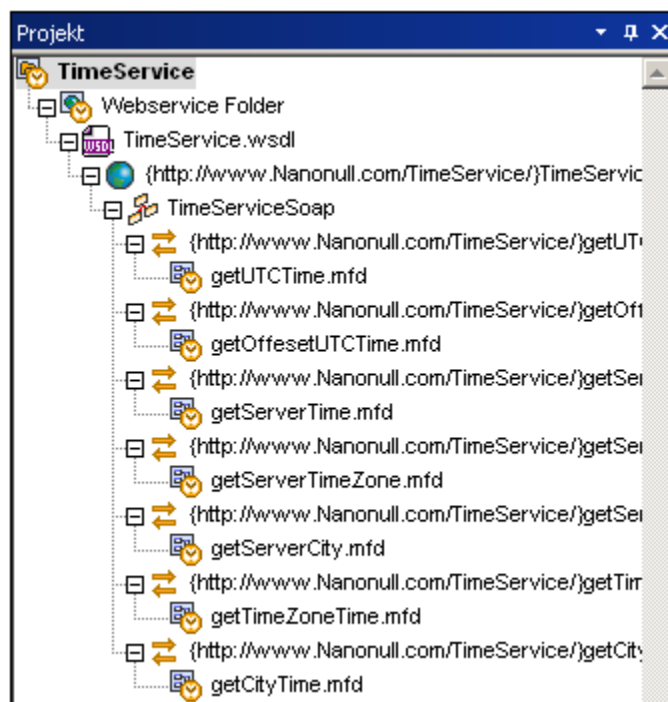
## 25.1 Aufruf der Webservice-Funktion getCityTime

In diesem Beispiel sehen Sie, wie ein Webservice, der selbst mit Hilfe von MapForce implementiert wurde, aufgerufen wird. Dies dient nur zu Demonstrationszwecken - der Webservice könnte auch mit jeder anderen Technologie implementiert werden, die ein kompatibles Protokoll unterstützt.

Das unten gezeigte Mapping ist Teil des Mapping-Projekts **TimeService.mfp**, das im Ordner **...MapForceExamples\TimeService** zur Verfügung steht. Die Projektdatei TimeService2.mfp liegt in einem separaten Ordner und unterstützt WSDL 2.0.

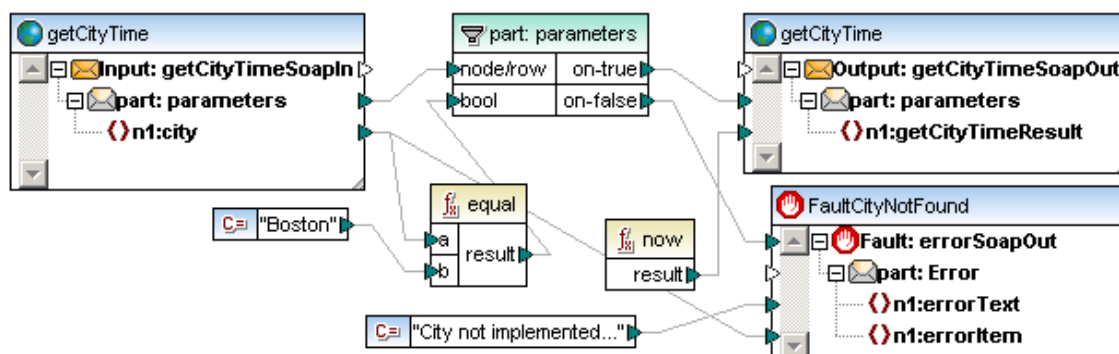
**Anzeige der vordefinierten Operationen/Mappings des TimeService Webservice-Projekts:**

1. Wählen Sie **Datei | Öffnen** und anschließend die Datei **TimeService.mfp** im Ordner **...MapForceExamples\Timeservice**.



Daraufhin wird die Projektdatei "TimeService" geöffnet und die Webservice-Operationen und die damit verknüpften \*.mfd -Mappingsdateien werden angezeigt. Beachten Sie: Diese Mapping-Dateien wurden für Sie vordefiniert.

2. Doppelklicken Sie im Projektfenster auf den Eintrag **getCityTime.mfd**.



In diesem Mapping wird der von der Konstante "Boston" bereitgestellte Wert mit dem von der Komponente **getCityTimeSoapIn** bereitgestellten Wert verglichen. Diese Komponente verwendet die Datei **getCityTimeRequest.xml** als Datenquelle für die Vorschau-Einstellungen.

Das oben gezeigte Mapping kann als Webservice-Funktion in eine Standard-Mapping-Datei eingefügt werden, ohne dass Sie dafür ein WSDL-Projekt erstellen müssen.

**So fügen Sie getCityTime als Webservice-Funktion ein:**

1. Wählen Sie **Datei | Neu**, klicken Sie auf das Mapping-Symbol und bestätigen Sie mit OK.
2. Wählen Sie die Menüoption **Einfügen | Webservice-Funktion...** oder klicken Sie in der Symbolleiste auf das  Symbol.
3. Klicken Sie auf die Schaltfläche "Durchsuchen", um die WSDL-Definitionsdatei auszuwählen; wählen Sie im TimeService-Verzeichnis **TimeService.wsdl** aus und klicken Sie auf die Schaltfläche "Öffnen".
4. Wählen Sie im nächsten Dialogfeld einen Webservice-Port aus z.B. **TimeServiceSoap** und klicken Sie auf OK.



5. Wählen Sie die Webservice-Operation **getCityTime** und klicken Sie zur Bestätigung auf OK.

**WSDL Call-Einstellungen**

**Globale Einstellungen**

WSDL-Definitionen:

Service:

Port:

Operation:

**Verbindungseinstellungen**

Endpoint:

**Authentifikation**

Benutzername:

Passwort:

In der Gruppe **Authentifikation** können Sie ggf. einen Benutzernamen und ein Passwort eingeben, um den Webservice aufrufen zu können.

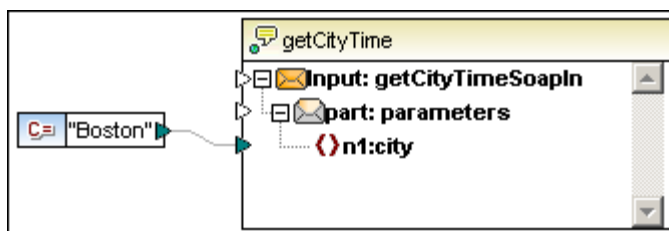
Die Webservice-Funktion getCityTime wird als einzelne Komponente eingefügt. Beachten Sie: Diese Komponente steht für alle acht Komponenten, aus denen die im WSDL-Projekt gespeicherte Datei **getCityTime.mfd** besteht.



Im linken Abschnitt der Komponente ist die Dateneingabe (SoapIn) definiert, während auf der rechten Seite die Datenausgabe (SoapOut) definiert ist. Dieser Abschnitt kann auch einen Fehler-Abschnitt enthalten, wenn in der WSDL-Datei einer definiert wurde.

#### So definieren Sie Input-/Output-Komponenten der WSDL-Funktion:

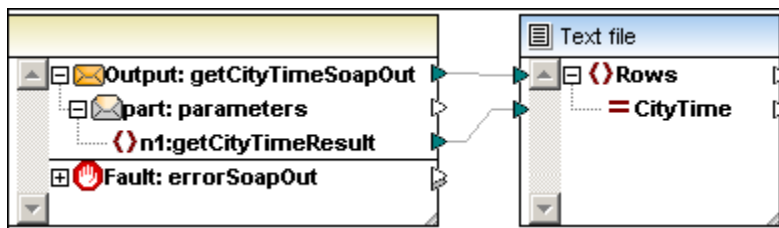
1. Fügen Sie die Komponente ein, die die Input-Daten bereitstellen soll, z.B. eine Konstante, Text oder eine Schemakomponente.  
Fügen Sie in diesem Fall eine Konstantenkomponente ein und geben Sie als Input String "Boston" ein.



2. Verbinden Sie die Konstante mit dem Datenelement **n1:city**.
3. Fügen Sie eine Textkomponente ein, geben Sie die Namen der Textdatei in das Feld "Ausgabedatei" ein und bestätigen Sie mit OK.

The screenshot shows the 'Text importieren / exportieren' dialog box. It has two input fields: 'Eingabe- / Ausgabedatei' (empty) and 'text.txt'. To the right of each field is a button labeled 'Eingabedatei' and 'Ausgabedatei' respectively.

4. Verbinden Sie n1:getCityTimeResult mit dem Feld in der Textkomponente.



5. Klicken Sie auf die Schaltfläche "Ausgabe" um die Mapping-Ausgabe zu sehen.

|   |         |
|---|---------|
| 1 | 5:56 AM |
| 2 |         |

Die aktuelle Zeit in Boston wird im Ausgabefenster angezeigt.

Bitte beachten Sie:

- Der Eingabewert der Webservice-Funktion hat immer Vorrang vor der Datenquelle des ursprünglichen Mappings. So hat z.B. der Wert der Konstantenkomponente "Boston" Vorrang vor der Datenquelle **getCityTimeRequest.xml** im Originalmapping.

Wenn die Input-Datei mehrere Einträge enthält, wird im Ausgabefenster für jedes Datenelement ein Ergebnis in einer eigenen Zeile angezeigt.

Z.B. eine Textdatei mit drei Einträgen:

| Cities list |  |
|-------------|--|
| string      |  |
| Boston      |  |
| Vienna      |  |
| New York    |  |
|             |  |

Ergebnis im Ausgabefenster:

|   |              |
|---|--------------|
| 1 | 6:30 AM      |
| 2 | 12:30 PM     |
| 3 | Unknown City |
| 4 |              |

### So mappen Sie Webservice-Fehler:

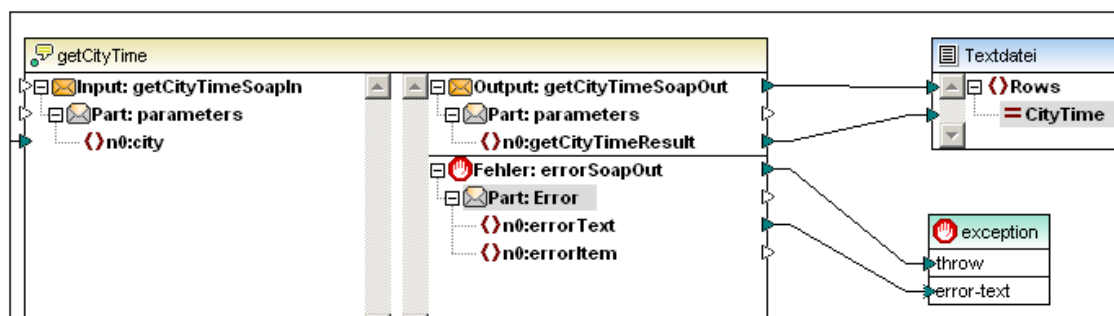
1. Wählen Sie den Befehl **Einfügen | Ausnahme** oder klicken Sie auf das Symbol "Ausnahmeereignis" .
2. Stellen Sie sicher, dass das Optionsfeld "Allgemeines Ausnahmeereignis erzeugen" aktiv ist, und klicken Sie zum Fortfahren auf OK.

**Select an exception**

☒ Create general exception  
☐ Create WSDL-fault

| Name              | Namespace-URI                        |
|-------------------|--------------------------------------|
| FaultCityNotFound | http://www.Nanonull.com/TimeService/ |

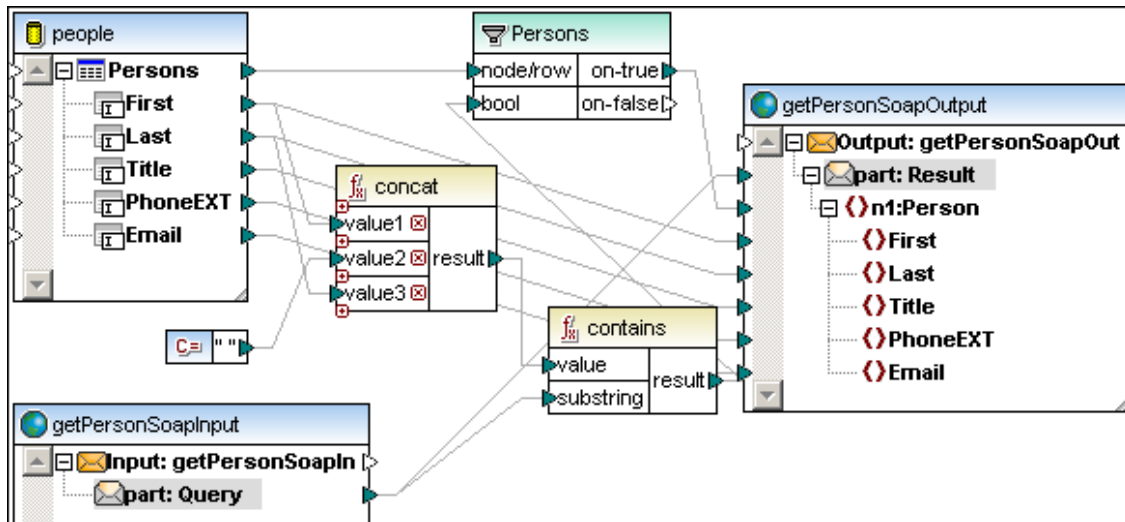
Daraufhin wird die Ausnahmeereigniskomponente eingefügt.



2. Mappen Sie das Datenelement **Fault:** auf das Datenelement **throw** der Ausnahmeereigniskomponente.
3. Mappen Sie das Datenelement **n1:errorText** auf das Datenelement **error-text** der Ausnahmeereigniskomponente.

## 25.2 Aufruf des Webservice getPerson

Das unten gezeigte Mapping ist Teil des Mapping-Projekts **Query Person Database.mfp** aus dem Ordner ...\\Tutorial. Eine Beschreibung dieses Webservice und eine Anleitung zum Generieren von Code dafür finden Sie im Abschnitt ["Erstellen von Webservice-Projekten anhand von WSDL-Dateien"](#).



In diesem Beispiel wird gezeigt, wie eine Webservice-Funktion eingefügt wird, um dasselbe Ergebnis zu erzielen, wie im Abschnitt "Erstellen von Webservice-Projekten" beschrieben. Es wird davon ausgegangen, dass der Webservice auf dem Webserver kompiliert und bereitgestellt wurde.

### So fügen Sie getPerson als eine Webservice-Funktion ein:

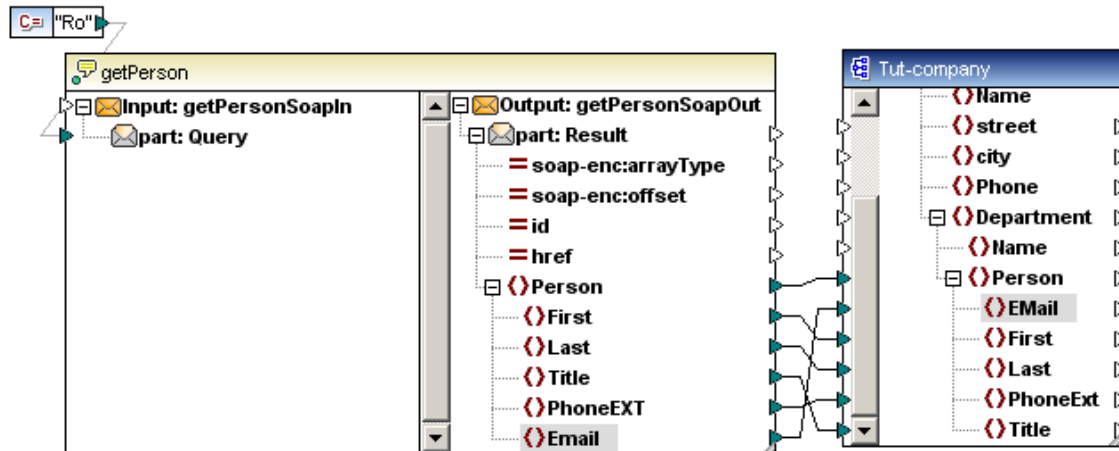
1. Wählen Sie den Befehl **Datei | Neu**, klicken Sie auf das Mapping-Symbol und bestätigen Sie mit OK.
2. Wählen Sie die Menüoption **Einfügen | Webservice-Funktion...** oder klicken Sie auf das  Symbol in der Symbolleiste.
3. Klicken Sie auf die Schaltfläche "Durchsuchen", um die WSDL-Definitionsdatei auszuwählen; wählen Sie **query.wSDL** aus dem Verzeichnis ...\\Tutorial aus und klicken Sie auf die Schaltfläche "Öffnen".
4. Wählen Sie im Dialogfeld "Webservice-Operation auswählen" den Eintrag **getPerson** aus und klicken Sie zur Bestätigung auf OK.



5. Fügen Sie eine Konstantenkomponente aus und geben Sie den String für die Abfrage der Datenbank aus, z.B. **Ro**.
6. Fügen Sie eine XML-Schemadatei ein, die die Ergebnisdaten enthalten soll, z.B. **Tut-company.xsd** aus dem Ordner ...\\Tutorial und stellen Sie eine Verbindung zu den



relevanten Datenelementen her.



7. Klicken Sie auf die Schaltfläche "Ausgabe", um die Ergebnisdaten zu sehen.



# Kapitel 26

---

## XBRL-Mapping

## 26 XBRL-Mapping

Altova Website:  [Mappen von XBRL-Dateien](#)

Das Altova Produktpaket unterstützt XBRL Version 2.1 und XBRL Dimensions Version 1.0, wobei jedes Altova-Produkte für einen anderen Aspekt von XBRL verwendet wird.

### XMLSpy 2012

dient zum Erstellen/Bearbeiten neuer Taxonomien (und generiert XBRL-Berichte anhand davon, wenn die XSLT-Transformationsdatei zur Verfügung steht - oder in StyleVision erstellt wurde)

### StyleVision 2012

dient zum Erstellen von Taxonomie Stylesheets/Vorlagen, mit Hilfe derer Sie XBRL-Berichte erstellen können.

### MapForce 2012

dient zum Mappen von Daten von/auf XBRL-Taxonomien oder Instanzdateien, wobei Excel 2007-Dateien, Datenbanken oder CSV-Dateien als Input/Output-Dokumente verwendet werden. Sie können auf diese Art Zwischenberichte erstellen oder bestimmte Daten aus XBRL-Instanzdokumenten filtern.

MapForce unterstützt das Mappen zwischen XBRL-Dokumenten, die auf den folgenden Taxonomien basieren:

- US-GAAP, einschließlich US-GAAP 2011
- IFRS
- COREP / FINREP

Beim Mappen von/auf XBRL-Komponenten wird davon ausgegangen, dass die zugrunde liegende XBRL-Taxonomiedatei verfügbar ist. Firmenspezifische Taxonomiedateien werden anhand einer bestimmten vorhandenen Taxonomie, z.B. us-gaap, die als Basis für die neue Taxonomie dient, erstellt. Die Basistaxonomie wird in der neuen Taxonomiedatei erweitert und es werden ein neuer Namespace und ein neues Namespace-Attribut und Präfix hinzugefügt.

### **Mappen einer Datenbank auf eine Taxonomie**

In der Beispieldatei [DB\\_to\\_XBRL.mfd](#) sehen Sie, wie Daten aus einer MS Access-Datenbank auf eine XBRL-Taxonomie gemappt werden, sodass eine gültige XBRL-Instanzdatei erzeugt wird.

### **Mappen eines Excel-Arbeitsblatts auf eine Taxonomie**

In der Beispieldatei [boa-balance-sheet.mfd](#) sehen Sie, wie Daten aus einem Excel-Arbeitsblatt auf eine Taxonomie gemappt werden, sodass eine gültige XBRL-Instanzdatei erzeugt wird.

## 26.1 Einfügen von XBRL-Dokumenten

Es gibt zwei Methoden, um XBRL-Dokumente in MapForce einzufügen:

- als einfaches flaches XML-Dokument
- als hierarchisches XBRL-Dokument

### Einfügen eines XML-Dokuments (XML)

Wählen Sie die Menüoption **Einfügen | XML/Schema-Datei einfügen**.

Daraufhin wird die XBRL-Datei als einfache XML-Datei ohne Business Logic-Konzepte oder hierarchische XBRL-Struktur eingefügt. Eine solche Komponente eignet sich nur für ganz einfache Mappings, für die keine Dimension-Behandlung oder automatische Generierung von XBRL Contexts erforderlich ist.



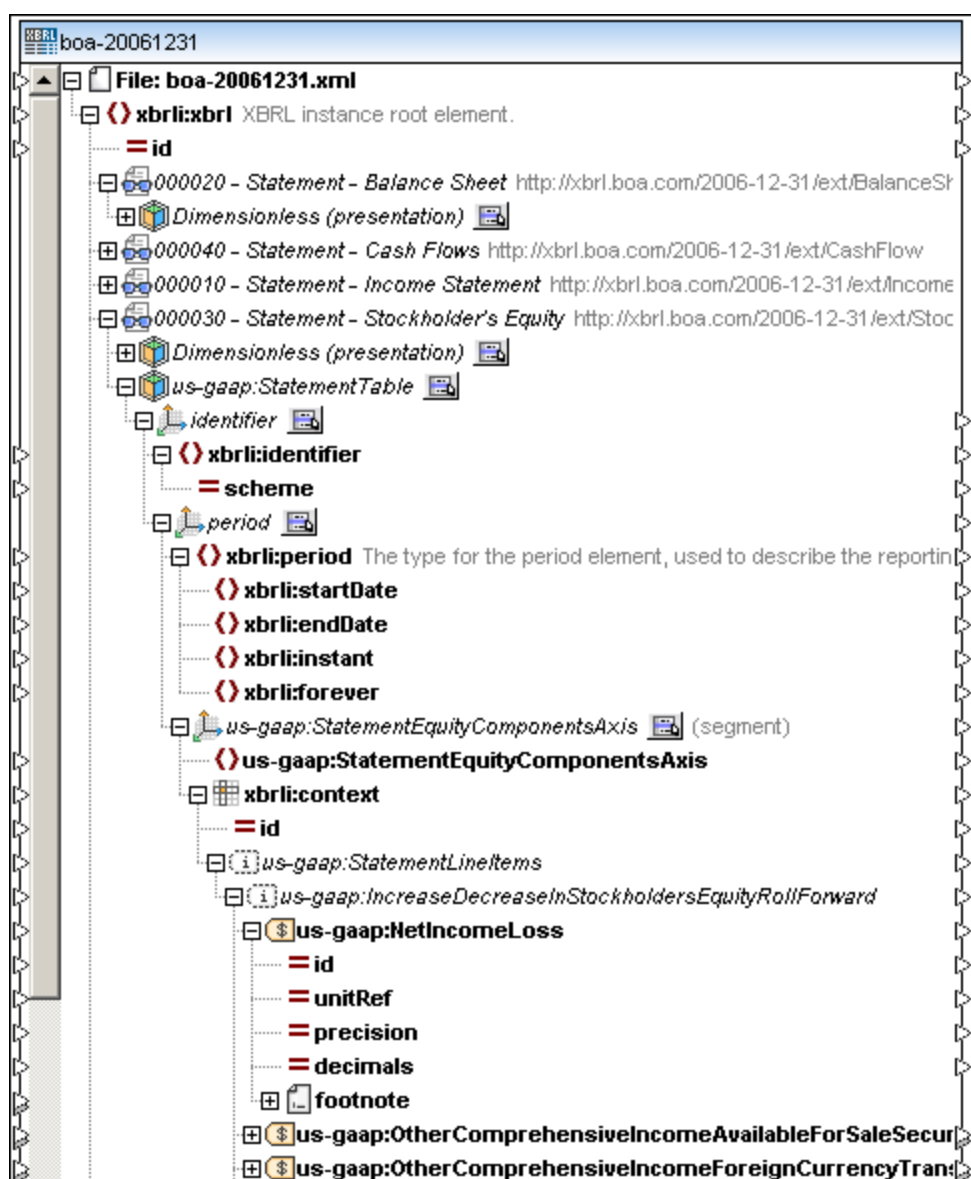
### Einfügen als ein XBRL-Instanzdokument / als XBRL-Taxonomie

Damit wird die XBRL-Taxonomie eingefügt und ihr Inhalt wird hierarchisch und XBRL-gerecht angezeigt.

1. Wählen Sie die Menüoption **Einfügen | XBRL-Dokument einfügen**
2. Wählen Sie die Taxonomiedatei (\*.xsd) oder eine XBRL-Instanzdatei (\*.xbrl oder \*.xml) aus: Falls Sie eine Taxonomie ausgewählt haben, werden Sie in einem weiteren Dialogfeld aufgefordert, eine gültige Instanzdatei auszuwählen.
3. Klicken Sie auf **Durchsuchen...**, falls Sie diese XBRL-Komponente als Quellinstanz verwenden möchten und wählen Sie die XBRL-Instanzdatei aus.

In der XBRL-spezifischen Struktur sehen Sie alle Berichte. Sie bietet beim Lesen und Schreiben von Hypercube Dimension-Daten und Aufrufen von Finanzbericht-Concepts, die in einer XBRL-Taxonomie definiert sind, automatische Context-Verwaltung. Die **Werte** dieser Elemente in einem Instanzdokument werden als **Facts** bezeichnet.

In der Abbildung unten sehen Sie eine typische XBRL-Hierarchie und eine Übersicht über alle Komponentenelemente, die normalerweise Teil einer XBRL-Komponente sind.



### XBRL-Komponentenelemente

|                     |                                                                                                                                                                                                                                                                                                                                                        |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| XBRL Root Element   | Das xbrli:xbrl Element ist das Instanz-Root-Element jeder XBRL-Komponente. Es enthält Berichte (Reporting statements), das xbrli:Unit-Element, das xbrli:context Element und die zwei Sonderelemente "Alle Concepts" und "Alle Concepts (nicht hierarchisch)", die beide über das Dialogfeld "XBRL-Komponenteneinstellungen" ausgewählt werden können. |
| Reporting statement | Reporting Statements (Berichte) stellen erweiterte Link Roles von der Definition und Presentation Linkbase einer XBRL-Taxonomie dar, z.B.<br><a href="http://xbrl.boa.com/2006-12-31/ext/CashFlow">http://xbrl.boa.com/2006-12-31/ext/CashFlow</a>                                                                                                     |

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hypercube           | Hypercubes gliedern Dimensions, Contexts und damit in Zusammenhang stehende XBRL-Concepts anhand von Informationen aus der Definition Linkbase und der Presentation hierarchisch. Es gibt verschiedene Arten von Hypercubes, z.B.  <i>Dimensionless (presentation)</i>  und  <i>Statement {Table}</i>  .                                                                                                         |
| Hypercube Dimension | Dimensions in XBRL dienen zum Gliedern von Context-Informationen für Business Facts. Dimensions sind in der Taxonomie innerhalb eines Hypercube z.B.  <i>Statement, Equity Components {Axis}</i>  ( <i>segment</i> ) definiert. Zusätzlich dazu werden in MapForce zwei Standard-Dimensions, nämlich  <i>identifier</i> und  <i>period</i> angezeigt. Dimensions sind entweder explicit oder typed Dimensions. |
| Dimension-Wert      | Dies sind die definierten Domain Member-Werte einer Dimension, z.B.:  <b>Retained Earnings [Member]</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Context             | Der Context Node  <b>xbrli:context</b> ist der Behälter für alle damit in Zusammenhang stehenden Business Facts. Context Nodes innerhalb von Hypercubes verwalten ihre Context-Elemente und Dimensions automatisch. Der xbrli:context Node, welcher ein Child des XBRL-Root-Elements ist, wird für die manuelle Behandlung von Dimensions verwendet.                                                                                                                                                                                                                                                                                                                  |
| Abstract item       | Abstract Items dienen zum Strukturieren von Facts. Sie sind entweder innerhalb der Presentation Linkbase oder innerhalb von Domain Member-Netzwerken von Definitions Linkbases wie z.B.  <i>Statement of Cash Flows {Abstract}</i> definiert. Sie sind nicht wirklich in der XBRL-Instanzdatei vorhanden. Mit Hilfe dieser Abstract Items können Facts auf intuitive Art gruppiert werden.                                                                                                                                                                                                                                                                          |
| Facts               | Facts sind die Werte von XBRL-Items. Sie können unterschiedliche Arten haben, z.B. monetary item type  <b>Net Income (Loss)</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Tuples              | Tuples sind komplexe Elemente, die Facts oder andere Tuples als Mitglieder enthalten, z.B.  <b>Accountants Report</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| unit                | Das Unit-Element  <b>xbrli:unit</b> enthält Units, auf die sich XBRL Items beziehen. Beim Mappen von Daten muss ein Wert für das xbrli:unit Element definiert sein, z.B. für Dollar UnitID = usd and Measure = iso4217:USD. Wenn Sie diesen Wert mittels einer Konstantenfunktion definieren, wird er auf alle Facts des XBRL-Instanzdokuments angewendet.                                                                                                                                                                                                                                                                                                          |
| footnote            | Mit Hilfe von Footnotes (Fußnoten)  <b>footnote</b> können Sie Concepts zusätzliche Textinformationen zuweisen.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### XBRL Item/Fact-Arten

In MapForce werden in der hierarchischen Struktur intuitive Symbole für die einzelnen Fact-Arten verwendet.

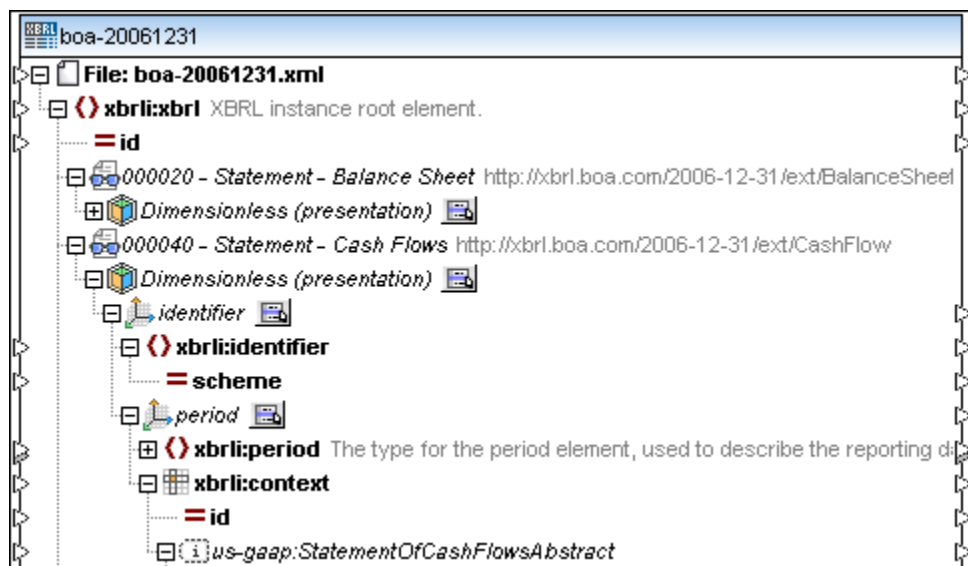
|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Monetary items |  <b>Debt Securities</b> xbrli:monetaryItemType                                                                                                                                                                                                                                                                                                                                                                |
| String items   |  <b>Accrual for Environmental Loss Contingencies, Caption</b> xbrli:stringItemType<br> <b>Accretion Expense [Member]</b> us-types:domainItemType<br> <b>Accounts Payable and Accrued Liabilities Disclosure [Text Block]</b> us-types:text  |
| Numeric items  |  <b>Debt Instrument, Convertible, Conversion Ratio</b> xbrli:decimalItemType<br> <b>Bankruptcy Claims, Number Claims Filed</b> xbrli:integerItemType<br> <b>Recorded Third-Party Environmental Recoveries, Discount Rate</b> us-types:perce |
| General items  |  <b>Adoption Date of SFAS 156</b> xbrli:dateItemType                                                                                                                                                                                                                                                                                                                                                          |
| Shares items   |  <b>Common Stock, Shares Outstanding</b> xbrli:sharesItemType                                                                                                                                                                                                                                                                                                                                                 |



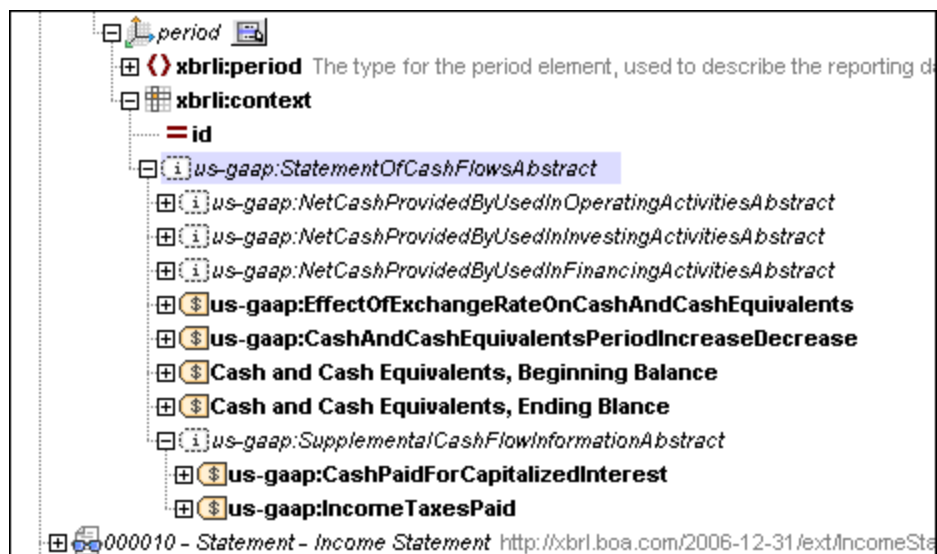
## 26.2 XBRL-Komponenten

In der Abbildung unten sehen Sie einen Ausschnitt aus einer XBRL-Datei, die folgende Elemente enthält:

- das xbrl-Root-Element
- Reporting Statements (Berichte) z.B. 00040 - Statement - Cash Flows
- einen "Dimensionless Hypercube" mit den Standard-Dimensions: Identifier und Period
- Abstract Elements z.B. *StatementOfCashFlowsAbstract*
- Concept/Fact-Elemente wie z.B. Effect of Exchange Rate Changes on Cash and Cash Equivalents.



Wenn Sie den Abstract Element-Eintrag *Statement Of Cash Flows Abstract* erweitern, sehen Sie weitere Abstract Elements sowie dunkelgelb angezeigte Items/Facts.

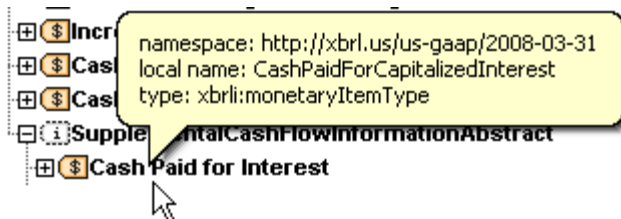


Wie Items/Facts in der Komponente dargestellt werden:

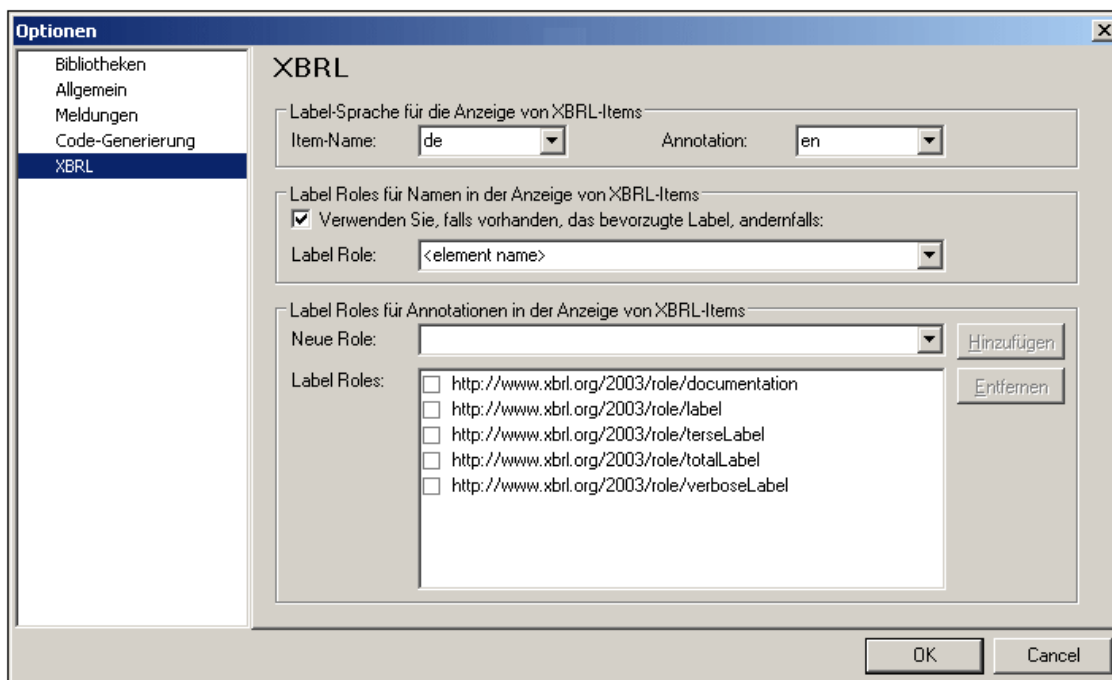
XBRL-Concept-Namen sind in der Taxonomie in der Label Linkbase definiert. MapForce kann

jede Label Role in der im Dialogfeld "XBRL-Anzeigeoptionen" konfigurierten Komponente oder dem nicht hierarchischen XML-Elementnamen anzeigen.

Beachten Sie: Wenn Sie den Mauszeiger über ein Abstract Element oder ein Fact platzieren, wird ein Popup mit zusätzlichen Informationen angezeigt: dem Namespace, dem lokalen Namen, z.B. **CashPaidForCapitalizedInterest** und dem Datentyp.



Globale XBRL Label-Optionen können durch Auswahl der Menüoption **Ansicht | XBRL-Anzeigeoptionen** definiert werden. Nähere Informationen dazu finden Sie unter [Ansicht](#).

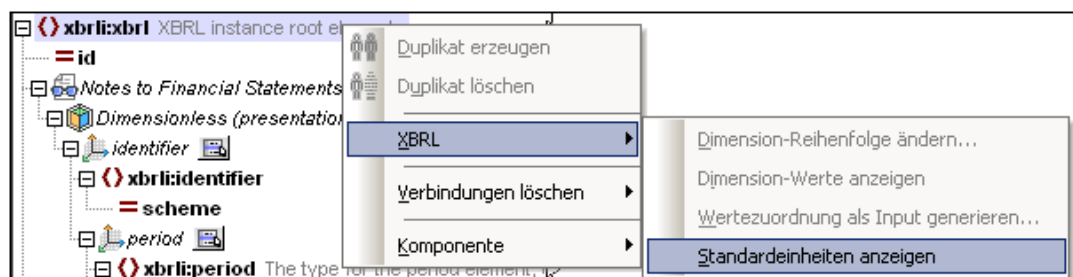


### Standardeinheiten

Default Units (Standardeinheiten) können für das gesamte XBRL-Dokument, Reporting Statements (Berichte), Hypercubes sowie Hypercube Dimensions aktiviert/definiert werden.

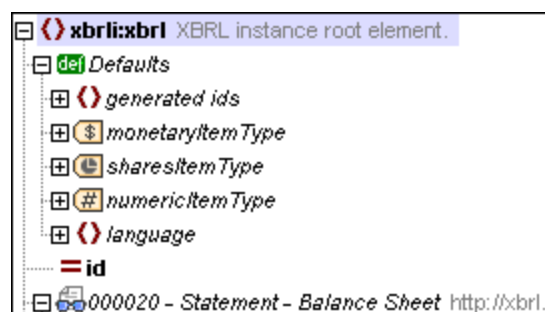
#### So werden Standardeinheiten angezeigt:

1. Klicken Sie mit der rechten Maustaste auf den Eintrag, für den Sie die Standardwerte definieren möchten, in diesem Fall auf das Root-Element und wählen Sie den Befehl **XBRL | Standardeinheiten anzeigen**.



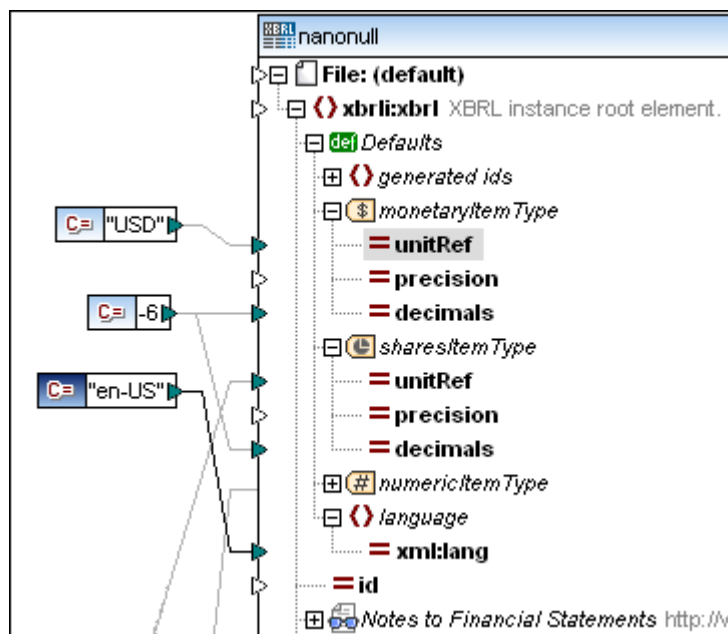
Beachten Sie: Sie können auch auf das Symbol  klicken und hier im Fall von Hypercubes und Hypercube Dimensions denselben Menübefehl auswählen.

Daraufhin wird ein Standardeintrag eingefügt, mit dem Sie Ihre eigenen Standardwerte für die verschiedenen Item-Arten verbinden können.



Standardwerte sind eine praktische Methode, um Attributen in der XBRL-Komponente Werte zuzuweisen, ohne dass Sie explizite Mapping-Verbindungen zu allen davon haben. Die Standardwerte werden für alle damit in Zusammenhang stehenden Attribute in der hierarchischen Substruktur verwendet.

Wenn Sie z.B. dem Attribut **unitRef** des **monetaryItem Type**-Elements in der Default-Hierarchie einen Constant Unit Identifier zuweisen, wird die Standardeinheit jedem Monetary XBRL Item Unit Identifier mit Ausnahme derer zugewiesen, bei denen die Input-Komponente explizit auf einen anderen Wert gemappt ist.



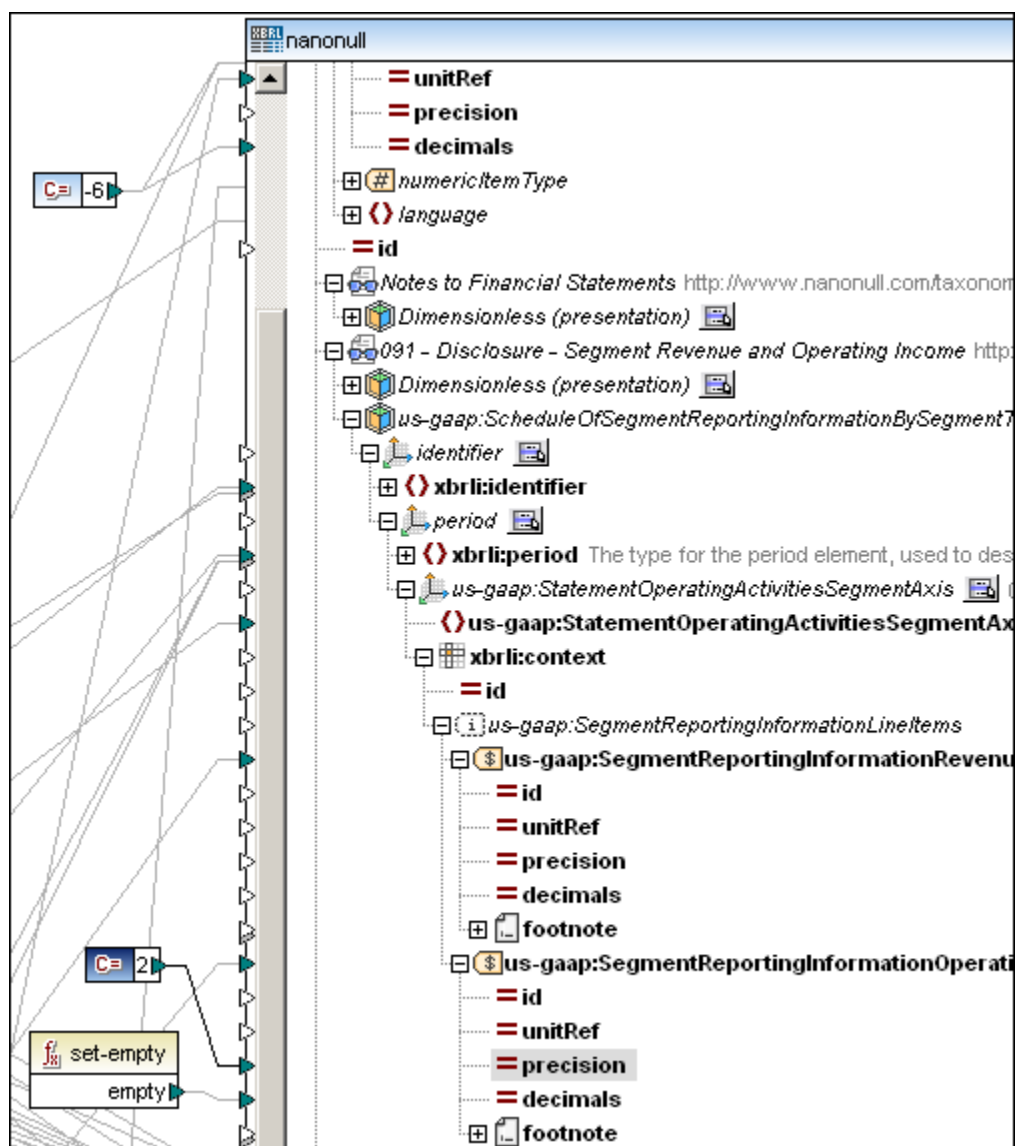
- Da für die verschiedensten Elemente der XBRL-Struktur Standardwerte definiert werden können, können unterschiedliche Substrukturen unterschiedliche Standardwerte haben.
- In der Abbildung sehen Sie auch die Verwendung des Standardwerts für das xml:lang Attribut, welches die Sprache einer Fußnote definiert.

### So ersetzen/deaktivieren Sie einen Standardwert

Wenn für einige Concept-Attribute, z.B. **decimals** ein Standardwert definiert wurde, so kann diese Einstellung mittels der Funktion **set-empty** lokal für einzelne Concepts entfernt werden.

In der folgenden Abbildung der Beispieldatei "DB\_To\_XBRL.mfd" deaktiviert die **set-empty** Funktion den Standardwert "-6" für das Monetary Item "Segment Reporting Information, Revenue".

Dieses Datenelement wird nun im Bericht mit dem Präzisionswert "2", der auf das precision-Attribut gemappt wurde, ausgegeben, während das andere Datenelement "Segment Reporting Information, Operating Income" mit dem Standardwert "-6" ausgegeben wird.



### Behandlung von Contexts

Die hierarchische Struktur in XBRL-Komponenten ermöglicht die Behandlung von Contexts. Die Generierung des **xbrli:context** in XBRL-Ausgabeinstanzen erfolgt automatisch bei der Ausgabe damit in Zusammenhang stehender Concepts.

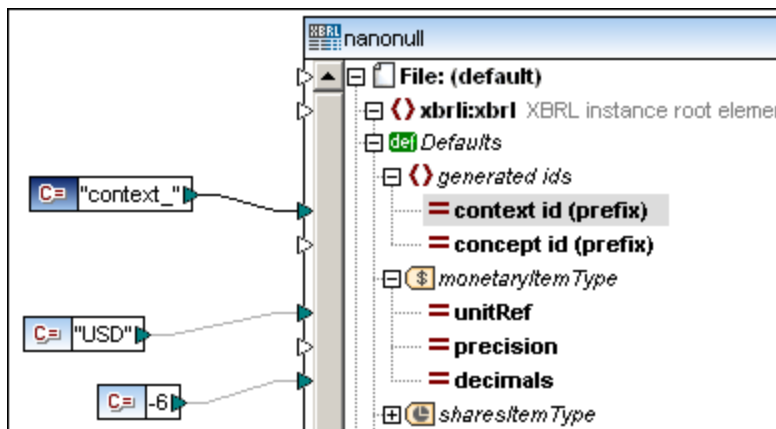
Der Wert des Attributs "id" eines bestimmten Context in einer XBRL-Instanz ist der Wert des Attributs "**contextRef**" in den einzelnen damit in Zusammenhang stehenden XBRL Concepts. MapForce nummeriert in einer Output-Instanz automatisch alle erstellten Contexts.

### So generieren Sie einen benutzerdefinierten Context:

Contexts lassen sich anpassen, indem Sie im Node "context id (prefix)" unter "generated ids" Text als ein Präfix zuweisen.

Wenn Sie z.B. den Konstantenwert "**context\_**" als Standardpräfix mappen, werden in der Ausgabeinstanz nacheinander Context-IDs mit den folgenden Werten erstellt: "context\_1", "context\_2", "context\_3", usw.

Hinweis: Wenn kein Standardwert definiert ist, erstellt MapForce alle Context-IDs mit dem Präfix "**ctx\_**".



Wenn die XBRL-Output-Instanz **Fußnoten** enthält, müssen die damit in Zusammenhang stehenden Concepts Concept-IDs haben, um mit den automatisch generierten Fußnoten-Links zu verlinken. Diese Attribute werden automatisch generiert. Das Datenelement "**concept id (prefix)**" kann gemappt werden, um ein solches Präfix zu definieren.

Hinweis: Wenn das Präfix nicht in der XBRL-Komponente gemappt ist, erstellt MapForce alle Concept-IDs mit dem Präfix "**fact\_**".

### Komponenten- / Anzeigeeinstellungen

Wenn Sie auf eine XBRL-Komponente doppelklicken, wird das Dialogfeld "Komponenteneinstellungen" geöffnet, das zusätzliche Anzeigeeinstellungen enthält, die definieren, welche Concepts in der Komponente angezeigt werden.

**Komponenteneinstellungen**

Komponentenname:

**Taxonomie**

**XBRL-Input-Datei**

**XBRL-Output-Datei**

Taxonomieschema-Referenz (leeres Feld für absoluten Pfad der Taxonomie):

☒ Werte in Zieltypen konvertieren (deaktivieren, um numerische oder Datumswerte beizubehalten, auf die Gefahr hin, dass eine ungültige Ausgabe erzeugt wird)

☒ Pretty Print für Ausgabe

☐ Digitale Signatur erstellen (nur Built-in-Ausführung)

Falls Signatur nicht erstellt werden kann: ☐ Verarbeitung stoppen ☒ Ohne Signatur fortfahren

**Kodierung**

Kodierungsname:

Bytefolge:  ☐ Bytefolge-Markierung inkl.

**Anzeigeeinstellungen**

☐ Eintrag "Alle Concepts" anzeigen

☐ Eintrag "Alle Concepts (nicht hierarchisch)" anzeigen

**StyleVision Power Stylesheet-Datei**

☒ Alle Dateipfade relativ zur MFD-Datei speichern

**Taxonomie:** Zeigt den Dateinamen und Pfad der Taxonomie-Hauptdatei an.

**XBRL-Input-Instanz:** Hier können Sie die XBRL-Instanz für die aktuell ausgewählte XBRL-Komponente auswählen oder wechseln. Dieses Feld wird ausgefüllt, wenn Sie zuerst die XBRL-Komponente einfügen und eine XBRL-Instanzdatei zuweisen.

**XBRL-Output-Instanz:** Dies ist der Dateiname und -pfad, in den die XBRL-Zielinstanzdatei

platziert wird, wenn Programmcode generiert und ausgeführt wird.

Der Eintrag aus dem Feld "XBRL-Input-Instanz" wird automatisch in das Feld kopiert, wenn Sie die XBRL-Instanzdatei zuweisen. Wenn Sie der Komponente keine XBRL-Instanzdatei zuweisen, enthält dieses Feld den Eintrag der Taxonomiedatei und die Erweiterung "xml".

**Taxonomieschema-Referenz:** Der Pfad des referenzierten/verknüpften Taxonomieschemas relativ zur MFD-Datei. Verwenden Sie dieses Feld, um einen anderen Taxonomiepfad für die Validierung zu definieren. Die Taxonomiereferenz wird in das href-Attribut des schemaRef Link-Elements geschrieben. z.B. <link:schemaRef xlink:type="simple" xlink:href="..\nanonull.xsd"/>.

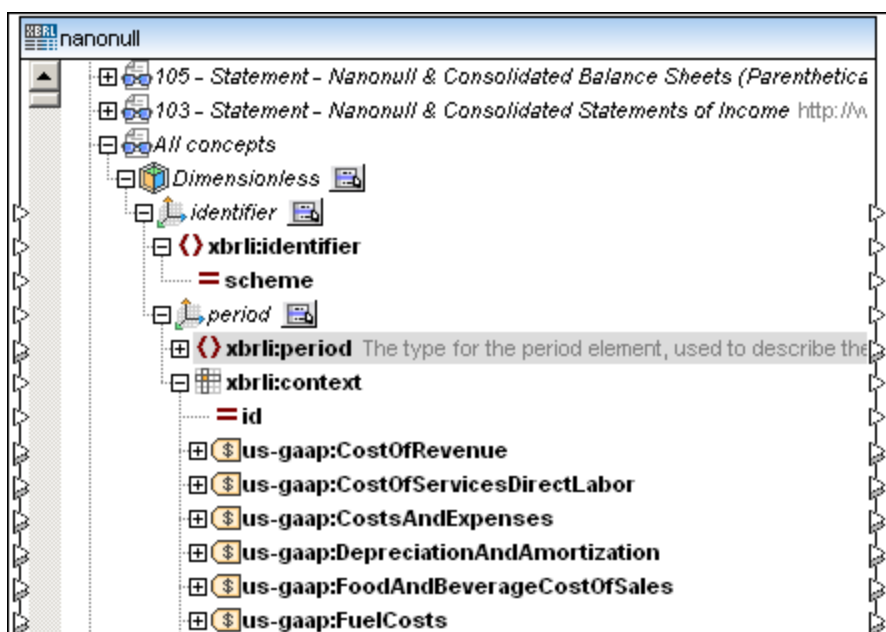
**Werte in Zieltypen konvertieren:** Hier können Sie definieren, ob beim Mapping die XML-Schemazieltypen (Standardeinstellung - aktiv) verwendet oder ob alle auf die Zielkomponente gemappten Daten als **Stringwerte** behandelt werden sollen.

**Pretty-Print für Ausgabe:** Formatiert Ihr XBRL-Dokument im Ausgabefenster neu, um eine übersichtliche Struktur zu erhalten. Jeder Child Node wird durch einen einzigen Tabulator von seinem übergeordneten Node eingerückt.

#### Kodierungseinstellungen

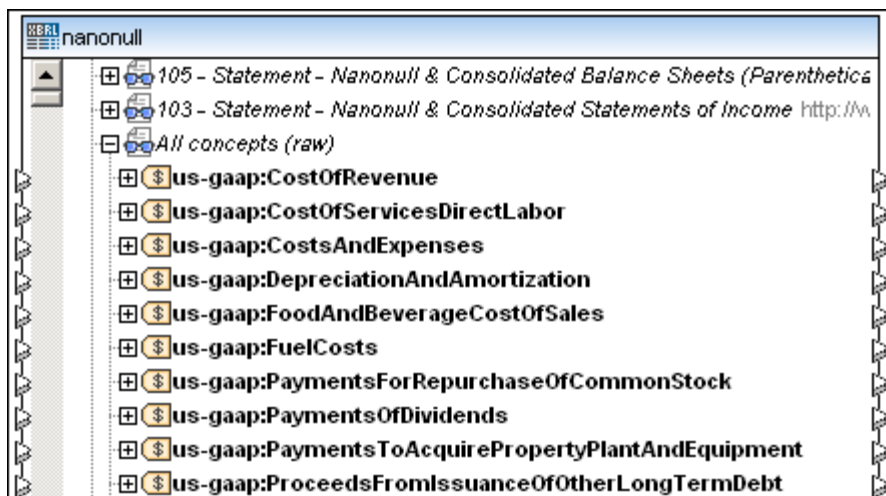
#### Eintrag "Alle Concepts" anzeigen:

Zeigt den zusätzlichen Node "Alle Concepts" in der hierarchischen Struktur an. Dieser Node enthält den Hypercube "Dimensionless", sodass alle Concepts der Taxonomie gemappt werden können, unabhängig davon, ob sie mittels der zwei Standard-Dimensions **identifizier** und **period** in Hypercubes inkludiert sind. Der **Context**-Node, der alle XBRL Concepts der Taxonomie als Subeinträge enthält, ermöglicht die automatische Behandlung von Contexts. Abstrakte Elemente (abstract items) werden nicht angezeigt.



#### Eintrag "Alle Concepts (nicht hierarchisch)" anzeigen:

Zeigt den zusätzlichen Node "Alle Concepts (nicht hierarchisch)" an, über den Sie Zugriff auf alle Facts der XBRL-Instanz haben, und zwar ohne Unterstützung für automatische Context- oder Dimension-Behandlung. Er enthält die nicht hierarchische XML-Struktur der Concepts in der Instanzdatei. Das **contextRef** Attribut muss beim Mappen dieser Datenelemente manuell gemappt werden.





### 26.2.1 XBRL-Hypercubes

Hypercube Items ermöglichen die automatische Behandlung von Contexts. Sie stellen XBRL-Hypercubes dar,

- die durch die Taxonomie definiert sind, z.B.  *Statement [Table]* 
- die von MapForce generiert wurden, um die Standard-Dimensions Identifier und Period aus der Presentation Linkbase zu vereinfachen, z.B.  *Dimensionless (presentation)* 
- die von MapForce innerhalb des "Alle Concepts" Node generiert wurden, z.B.  *Dimensionless*

Jeder Hypercube enthält zwei **Standard-Dimensions**, Identifier und Period, die das einfache Lesen/Schreiben dieser beiden Elemente für die einzelnen Contexts unterstützen. Zusätzliche in der Taxonomie definierte Dimensions werden automatisch mit den Context-Elementen xbrli: **segment** und xbrli: **scenario** in Beziehung gesetzt.

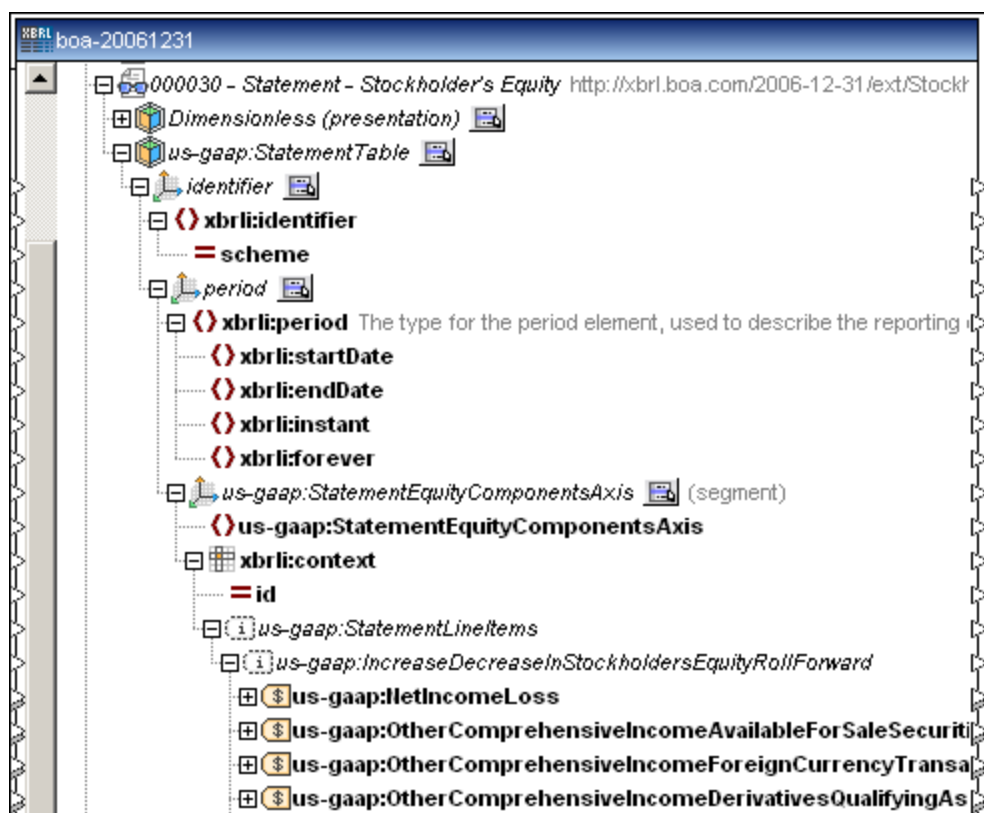
In als "**Dimensionless (presentation)**" gekennzeichneten Hypercubes werden beide Standard-Dimensions verwendet. Die hierarchische Reihenfolge der Concepts, die in ihrem Context-Node angezeigt werden, stammt aus der Presentation Linkbase.

In "**Dimensionless**" Hypercube-Elementen werden ebenfalls beide Standard-Dimensions verwendet, es gibt jedoch keine hierarchische Concept-Reihenfolge und es wird nur die nicht hierarchische Liste aller in der Taxonomie definierten Concepts angezeigt. Nähere Informationen dazu finden Sie unter [Komponenteneinstellungen](#).

Alle anderen Hypercubes sind in der Taxonomie definiert und werden entsprechend dem in der Label Linkbase der Taxonomie definierten Namen angegeben.

XBRL-Hypercubes und ihre Dimensions werden entweder als "Dimensionless" angezeigt, wenn keine Hypercube Dimensions verwendet werden, oder sie werden mit dem Hypercube-Namen z.B. Statement [Table] angezeigt. Hypercubes haben zwei Standard-Dimensions: Identifier und Period, welche bei beiden Hypercube-Arten sichtbar sind.

Hypercubes sowie ihre Dimensions (oder Axis) verfügen jeweils über ein kleines Symbol  über das Sie ein Popup-Menü aufrufen können, in dem Sie die Darstellung (Presentation) für die einzelnen Dimensions in der Komponente definieren können. Im unten gezeigten Diagramm sehen Sie die Standardansichtseinstellungen, beim Öffnen einer Taxonomiedatei.



Hinweis: MapForce zeigt alle Hypercubes an, die Concepts haben, die in Berichten ausgegeben werden können. Wenn eine der damit verknüpften Hypercube Dimensions keine Domain hat, wird sie in der XBRL-Komponente nicht angezeigt.

### Dimensions

Dimension-Elemente in XBRL beziehen sich entweder auf **explizite (explicit)** oder **typed** Dimension-Werte in der Instanz. Die Annotationen der einzelnen Dimension-Elemente zeigen in Klammern an, ob die Dimension in den Context-Elementen xbrl:segment oder xbrl:scenario angegeben wird.

Bei **Typed** Dimension-Elementen werden die Elemente ihres XML-Schema-Typs als Child-Elemente angezeigt. Ihre Werte können direkt gemappt werden.

**Explicit** Dimensions in einer XBRL-Taxonomie haben einen Wert vom Typ xs:QName aus einer bestimmten Domain. Diese umfasst die Werte der Mitglieder der XBRL Domain und die Werte des XBRL Domain-Elements selbst.

Explicit Dimensions können, je nach Mapping-Anforderungen und abhängig von der anderen Komponente/Struktur, auf die von/auf XBRL gemappt wird, auf zwei verschiedene Arten angezeigt werden.

### Wie Dimensions in einer Komponente angezeigt werden

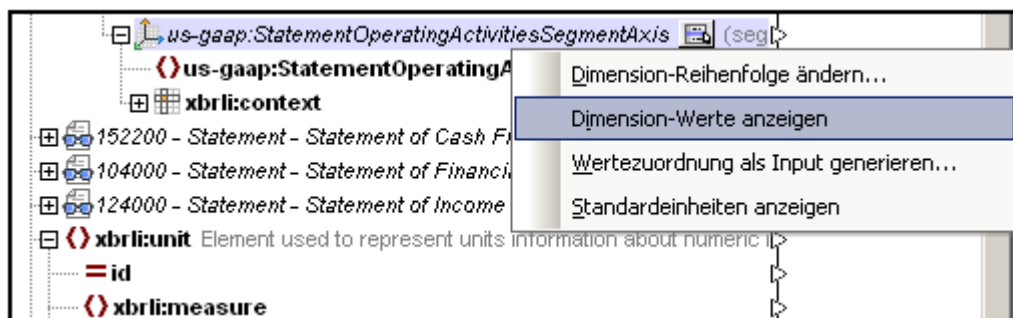
Anfangs wird die explicit Dimension mit einem **einzigen** Child Node angezeigt und kann direkt über dieses Child-Element gemappt werden, z.B. "Statement, Equity Components [Axis]".

Dies ist (für eine XBRL-Zielkomponente) nützlich, wenn die Dimension-Werte von einem Feld der Quelldaten z.B. einem Datenbankfeld oder einer Spalte in einer Excel-Tabelle ermittelt werden können. Da die Quelldaten im Allgemeinen den erforderlichen QName-Datentyp nicht enthalten, kann MapForce sie automatisch mit Hilfe der [value-map Funktion](#) erstellen.

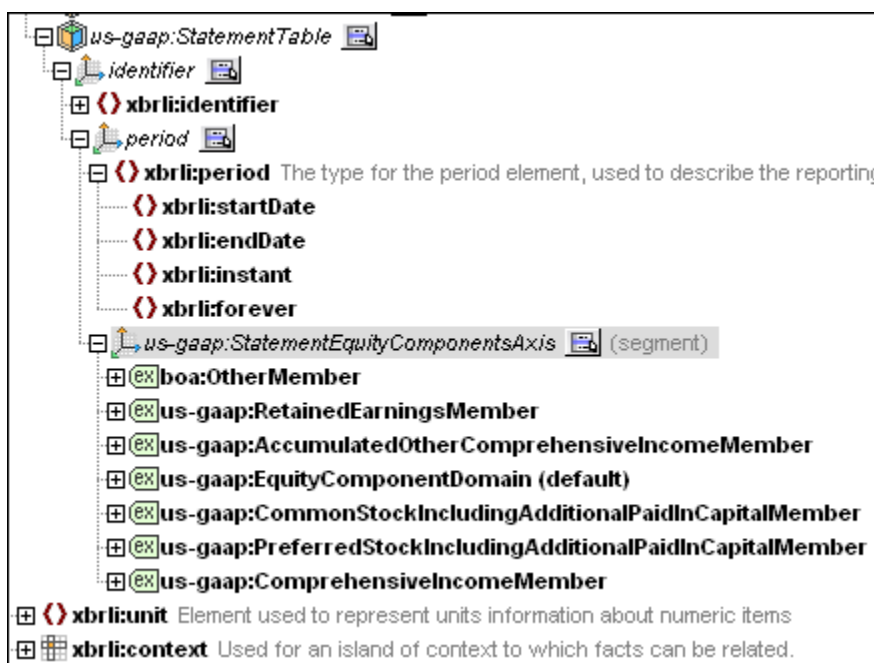
Um für die Facts unterschiedliche Mappings zu ermöglichen, können separate Nodes für jeden einzelnen Wert der Dimension Domain angezeigt werden.

#### So zeigen Sie Dimension-Werte in der Komponente an:

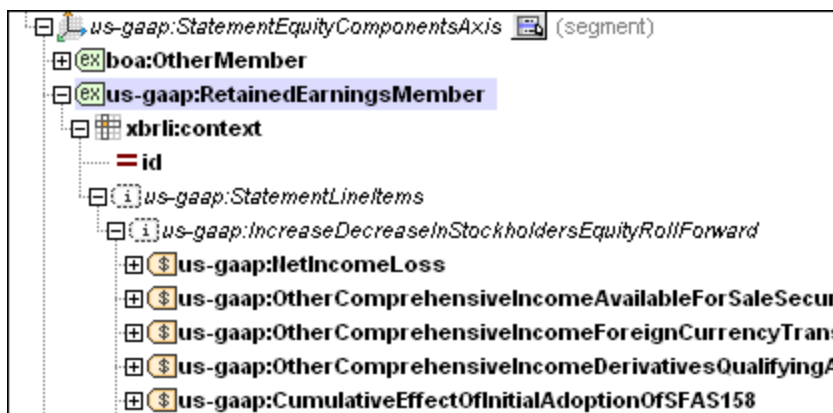
1. Klicken Sie auf das Symbol  der Dimension, für die die Werte angezeigt werden sollen, und wählen Sie den Befehl "Dimension-Werte anzeigen".



Daraufhin ändern sich die Einträge, die Sie unterhalb des Dimension-Namens sehen. Die **Dimension**-Domain und die **Member** Items sind nun sichtbar, wobei jeder dieser Einträge mit einem hellgrünen Symbol gekennzeichnet ist. Alle diese Einträge sind explizite Mitglieder (explicit members) einer Domain, was durch das Präfix "ex" im Symbol für den Eintrag gekennzeichnet ist.



Jedes explicit Member enthält nun dieselbe Substruktur, sodass für jedes andere Mappings möglich sind.



Wenn der Output eines Concept gemappt wird, werden nur jene Werte verwendet, für die das dazugehörige Context-Element den entsprechenden Dimension-Wert hat. So wird z.B. der Wert "Net Income (Loss)" in der Instanz nur für Contexts gemappt, die den Dimension-Wert "Comprehensive Income [Member]" für die Dimension "Statement, Equity Components [Axis]" enthalten. Es ist keine zusätzliche Filterung erforderlich.

Beim Schreiben von XBRL-Output-Instanzen wird die automatische Generierung ordnungsgemäßer Dimension-Werte im Context unterstützt. So erhält z.B. der Context Node `xbrli:context` für jedes im Bericht ausgegebene Monetary Item ein Element für die explizite Dimension "Statement, Equity Components [Axis]", das den Wert "Comprehensive Income [Member]" enthält.

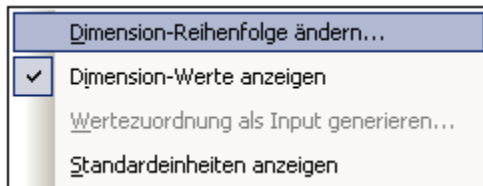
### Dimension-Reihenfolge

Anfangs werden alle Dimensions eines Hypercube in MapForce als verschachtelte Child Nodes angezeigt, wodurch automatisch eine Hierarchie entsteht. Die hierarchische Reihenfolge innerhalb des Hypercube kann geändert werden, sodass sie der anderen Komponente des Mappings (der nicht XBRL-Komponente) entspricht.

Wo Dimension-Werte für einige Concepts spezifisch definiert werden müssen, kann MapForce eine Dimension ohne Hierarchie anzeigen und diese als Child-Element des Context Node anzeigen.

### So ändern Sie die Reihenfolge der Dimension-Einträge:

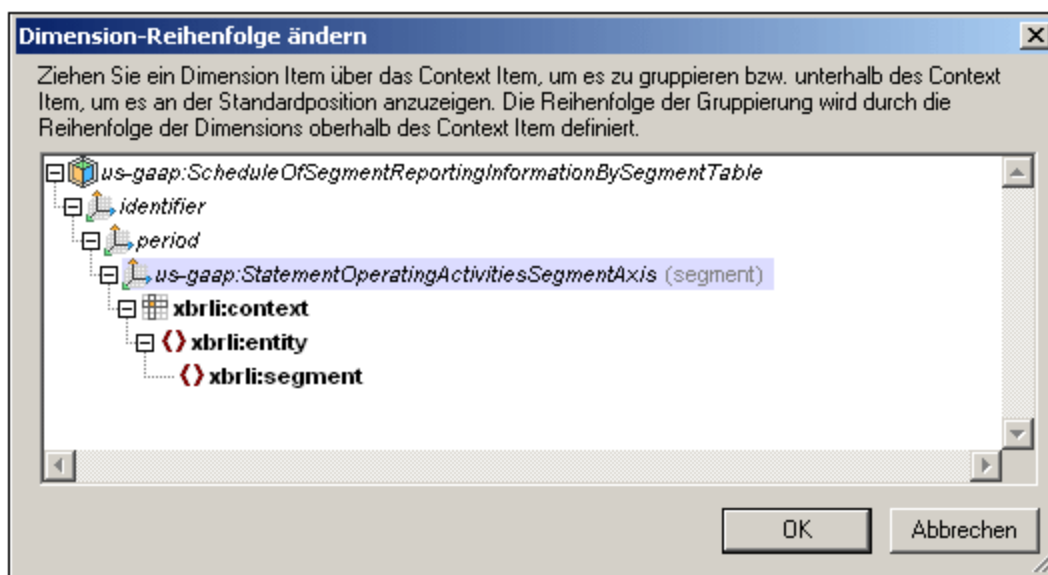
1. Klicken Sie auf ein beliebiges -Symbol des entsprechenden Hypercube und wählen Sie im Popup-Fenster den Befehl "Dimension-Reihenfolge ändern".



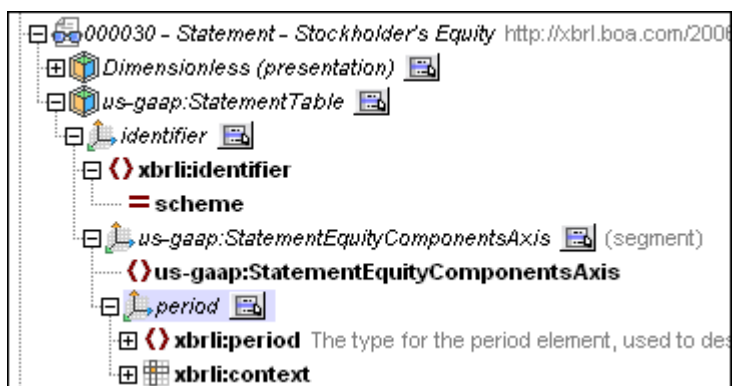
Daraufhin wird ein Dialogfeld geöffnet, in dem Sie die verschiedenen Dimensions eines Hypercube neu positionieren können.

Beachten Sie, dass ein Hypercube zwei **Standard-Dimensions** hat, nämlich Identifier und Period, deren Reihenfolge im Hypercube ebenfalls geändert werden kann.

2. Klicken Sie auf die Hypercube-Dimension und ziehen Sie sie mit der Maus an die gewünschte Stelle im Dialogfeld.



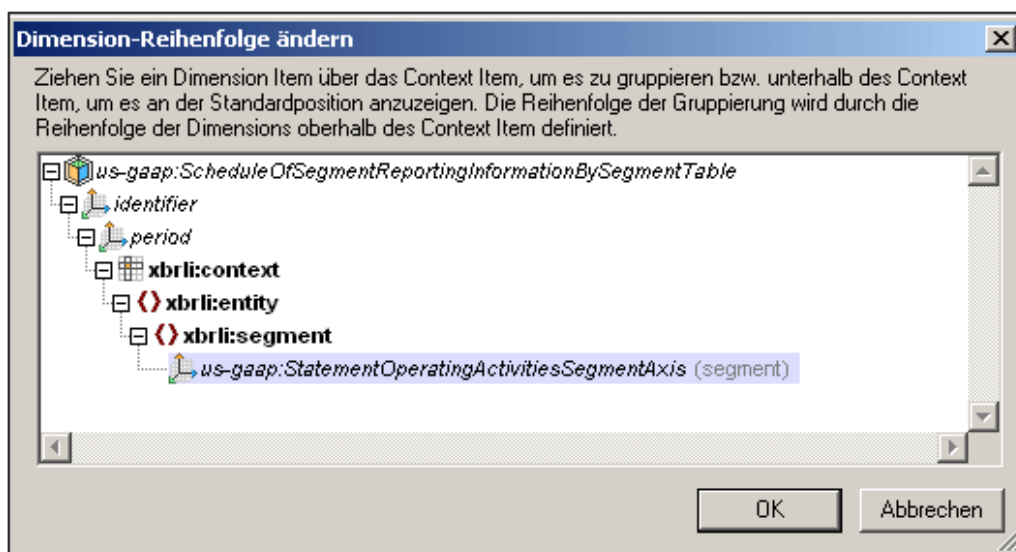
An Stellen, an die Sie die Dimension ziehen können, erscheint eine Linie.



3. Klicken Sie auf OK, um das Dialogfeld zu schließen, damit die Dimension im Dokument an die neue Position verschoben wird.

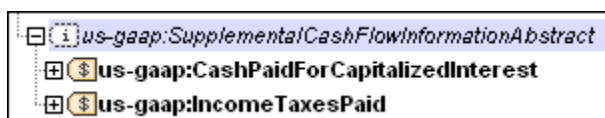
So **exkludieren** Sie eine Dimension aus einer Hierarchie:

- Ziehen Sie die Dimension unter die Zeile **xbrli:context**. Daraufhin wird sie in dieses Context Item eingefügt.



### Wie die MapForce-Hierarchie anhand der Taxonomie erstellt wird

Die **hierarchische** Struktur, die Sie in der Komponente sehen, wird durch die **xlink:from ... xlink:to** Presentation Arcs in der Presentation Linkbase erstellt.



Links von den Concepts zu den verschiedenen Facts:

```
<link:presentationArc xlink:type="arc" xlink:arcrole="http://www.xbrl.org/2003/arcrole/parent-child"
```

```
xlink:from="
```

```
SupplementalCashFlowInformationAbstract"
```

```
 xlink:to="CashPaidForCapitalizedInterest"
```

```
xlink:from="
```

```
SupplementalCashFlowInformationAbstract"
```

```
 xlink:to="IncomeTaxesPaid"
```

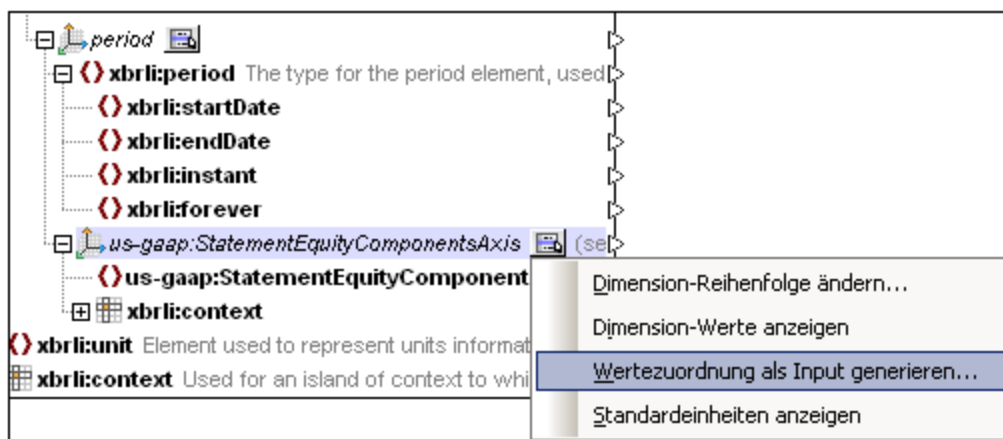
## 26.2.2 XBRL-Wertezuordnungsfunktion

Mit dieser Option werden Input-Daten jedes Typs in der Zielkomponente in einen gültigen QName transformiert, d.h. der Input-String wird in den Namen mit Präfix (QName) umgewandelt.

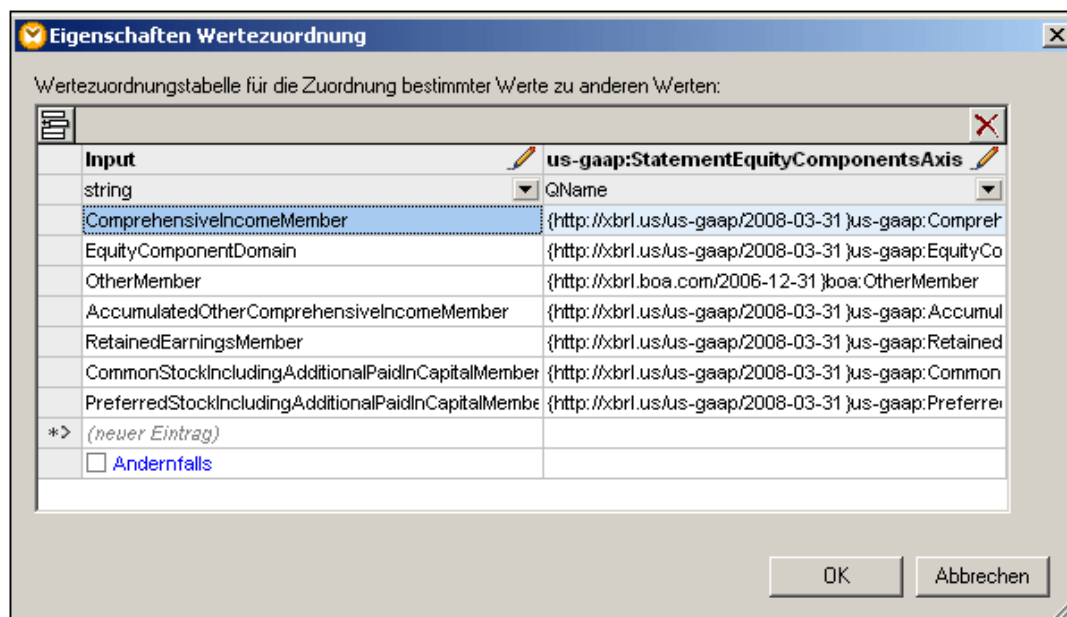
### So generieren Sie eine Wertezuordnungstabelle für die ausgewählte Hypercube-Dimension:

Stellen Sie sicher, dass die Dimension-Werte für die entsprechende Hypercube-Dimension nicht sichtbar sind.

1. Klicken Sie auf das Symbol  der entsprechenden **Dimension** und wählen Sie im Popup-Fenster die Option "Wertezuordnung als Input generieren".



Daraufhin wird das Dialogfeld "Eigenschaften Wertezuordnung" geöffnet, das automatisch anhand der Standard-Dimension Domain und der laut Taxonomie definierten Domain Members generierte Input- und Output-Werte enthält.



2. Bearbeiten Sie die Input-Werte, falls erforderlich, und klicken Sie auf OK, um die

Wertezuordnungskomponente einzufügen. Beachten Sie, dass Sie den Text der Spaltenüberschriften durch Doppelklick auf die Überschrift bearbeiten können, um ihn zu kürzen oder eine bessere Beschreibung einzugeben.

Daraufhin wird eine Wertezuordnungsfunktion (value-map) eingefügt, in der die Namen des Input- und Output-Parameters angezeigt werden. Der Output-Konnektor wird automatisch mit dem Domain-Element der Zielkomponente verbunden.



3. Verbinden Sie das Quelldatenelement, das die zu transformierenden Input-Daten enthält, mit dem Input-Parameter der Funktion.  
Der Input-String "USA" wird nun für das explizite Mitglied dieser Dimension in nanonull:USA transformiert.

Ein allgemeines Beispiel zur Verwendung der Wertezuordnungsfunktion finden Sie unter [Wertezuordnung - Transformieren von Input-Daten](#).



## 26.3 Beispiel Excel - XBRL

Die in den folgenden Beispielen verwendete Mapping-Datei finden Sie unter dem Namen **boa-balance-sheet.mfd** im Ordner ...\\MapForceExamples\\Tutorial\\. Die im Beispiel verwendeten Taxonomiedateien stammen von der Website [www.xbrl.us](http://www.xbrl.us) (klicken Sie auf den Link "Download all files as ZIP") und befinden sich auch im Ordner ..\\Tutorial.

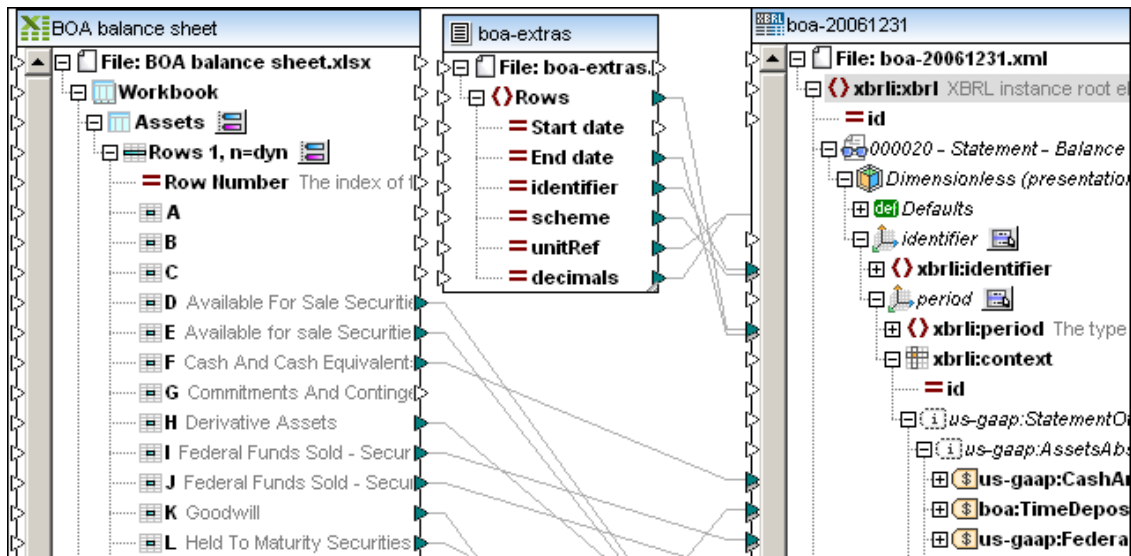
In dem Beispiel sehen Sie, wie Daten aus einem Excel-Arbeitsblatt auf eine XBRL-Taxonomie gemappt werden, um eine XBRL-Instanzdatei zu generieren. Um das Beispiel einfach zu halten, wurden nur Daten aus dem Arbeitsblatt "Assets" gemappt. Das Ergebnis des Mappings ist ein gültiges XBRL-Instanzdokument, das die Assets-Daten für eine bestimmte Instanz, den 31. Dezember 2006 enthält.



### 26.3.1 Beschreibung des Beispiels

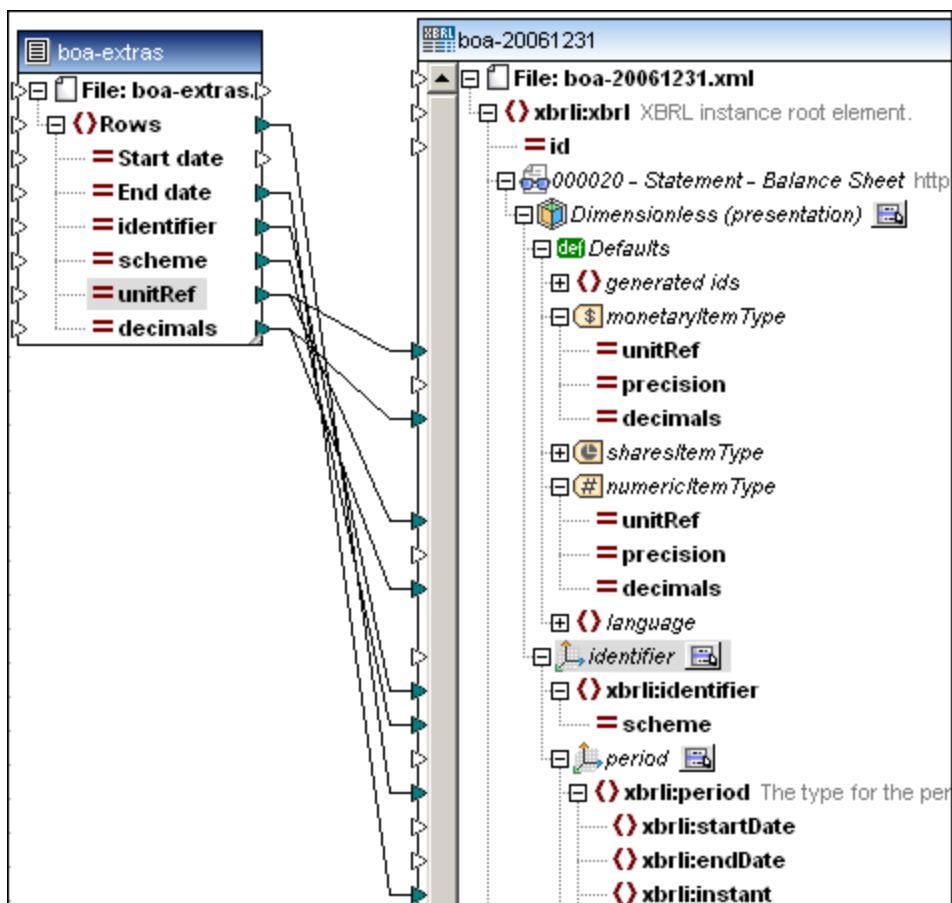
In diesem Beispiel sehen Sie, dass die Excel-Spalten, die in alphabetischer Reihenfolge angeordnet sind, in der Taxonomiekomponente direkt auf die XBRL Items/Facts wie z.B. "Cash and Cash Equivalents" gemappt werden.

Beachten Sie, dass die Namen der Datenelemente in der Quell- und Zielkomponente nicht identisch sind und auch nicht identisch sein müssen. In der erzeugten XBRL-Instanzdatei wird als Name der Name der Zieltaxonomie verwendet. Bei identischen Namen des Quell- und Zieldatenelements haben Sie allerdings den Vorteil, dass auch Sub-Einträge desselben Namens mit Hilfe der Funktion [Sub-Einträge automatisch verbinden](#) automatisch verbunden werden.



Um eine gültige XBRL-Instanzdatei generieren zu können, muss eine ganze Reihe von Datenelementen gemappt werden. Einige obligatorische Datenelemente (Items) stammen aus einer Textdatei (boa-extras), deren Felder auf Sub-Elemente des Datenelements "Defaults" im Dimensionless (Presentation) Hypercube gemappt werden.

Das Datenelement "Defaults" wird durch Rechtsklick auf eine Hypercube Dimension und Auswahl des Befehls **XBRL | Standardeinheiten anzeigen** oder durch Klicken auf das Symbol  und Auswahl der entsprechenden Option eingefügt.



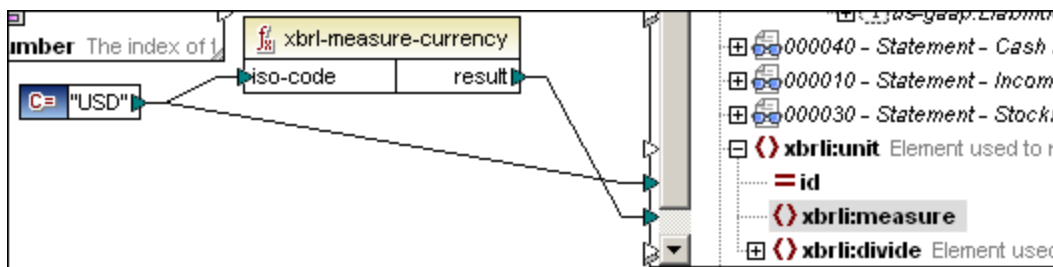
Obligatorische XBRL-Items, die in einer XBRL-Instanzdatei benötigt werden:

- **unitRef** und entweder **decimals** oder **precision** in Monetary Concepts.
- **xbrli:identifier** und **scheme** der Identifier Dimension
- **xbrli:period** und entweder **xbrli:instant** oder die Elemente **xbrli:startDate**/**xbrli:endDate**
- **xbrli:id** und **xbrli:measure** im Element **xbrli:unit**

Aus der Textdatei "boa-extras" stammen die Daten für einige der obligatorischen Datenelemente, deren Werte Sie unten sehen.

| CSV Einstellungen                                                                      |                                            |                             |                                                                          |                                                         |
|----------------------------------------------------------------------------------------|--------------------------------------------|-----------------------------|--------------------------------------------------------------------------|---------------------------------------------------------|
| Feldbegrenzung:                                                                        |                                            |                             |                                                                          |                                                         |
| <input type="radio"/> Tab                                                              | <input checked="" type="radio"/> Semikolon | <input type="radio"/> Komma | <input type="radio"/> Leerz.                                             | <input type="radio"/> Benutzerdef. <input type="text"/> |
| Textbegrenzung:                                                                        |                                            |                             |                                                                          |                                                         |
| <input type="radio"/> Keine <input type="radio"/> ' <input checked="" type="radio"/> " |                                            |                             |                                                                          |                                                         |
| <input type="radio"/> CSV <input type="radio"/> Feste Länge                            |                                            |                             |                                                                          |                                                         |
| <input type="checkbox"/> Erste Zeile enthält Feldnamen                                 |                                            |                             |                                                                          |                                                         |
| <input checked="" type="checkbox"/> Leere Felder als nicht vorhanden behandeln         |                                            |                             |                                                                          |                                                         |
| Start date                                                                             | End date                                   | identifier                  | scheme                                                                   | unitRef                                                 |
| date                                                                                   | date                                       | string                      | string                                                                   | string                                                  |
| 2006-01-01                                                                             | 2006-12-31                                 | 0000070858                  | http://www.sec.gov/Archives/edgar/data/70858/000119312507042036/d10k.htm | USD                                                     |

Die Daten für die Datenelemente "id" und "measure", die sich an der Basis der Taxonomiekomponente befinden, werden durch eine Konstantenkomponente geliefert.



Wenn Sie auf die Schaltfläche "Ausgabe" klicken, sehen Sie die erzeugte XBRL-Instanzdatei.

- Durch Klicken auf die Schaltfläche "Ausgabe validieren"  in der Ausgabesymbolleiste, können Sie die Gültigkeit der XBRL-Instanz überprüfen. Meldungen oder Warnungen werden im Fenster "Meldungen" angezeigt.
- Durch Klicken auf die Schaltfläche "Einstellungen für Textansicht"  können Sie die Einstellungen für die Ausgabeansicht definieren.

```
<?xml version="1.0" encoding="UTF-8"?>
<xbrl xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xl="http://www.xbrl.org/2003/instance" >
 <link schemaRef xlink:type="simple" xlink:href="A:/AA-tutorial-xbrl-example/boa-20061231.xsd" />
 <context id="ctx1">
 <entity>
 <identifier scheme="http://www.sec.gov/Archives/edgar/data/70858/000119312507042036/d10k.htm">0000070858</identifier>
 </entity>
 <period>
 <instant>2006-12-31</instant>
 </period>
 </context>
 <us-gaap:CashAndCashEquivalentsAtCarryingValue contextRef="ctx1" unitRef="USD" decimals="-6">36429</us-gaap:CashAndCashEquivalentsAtCarryingValue>
 <boa:TimeDepositsPlacedAndOtherShorttermInvestments contextRef="ctx1" unitRef="USD" decimals="-6">13952</boa:TimeDepositsPlacedAndOtherShorttermInvestments>
 <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResell contextRef="ctx1" unitRef="USD" decimals="-6">8</us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResell>
 <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResellPledgedAsCollateral contextRef="ctx1" unitRef="USD" decimals="-6">153052</us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResellPledgedAsCollateral>
 <us-gaap:TradingSecuritiesOther contextRef="ctx1" unitRef="USD" decimals="-6">922274</us-gaap:TradingSecuritiesOther>
 <us-gaap:TradingSecuritiesPledgedAsCollateral contextRef="ctx1" unitRef="USD" decimals="-6">23439</us-gaap:TradingSecuritiesPledgedAsCollateral>
 <us-gaap:DerivativeAssets contextRef="ctx1" unitRef="USD" decimals="-6">192806</us-gaap:DerivativeAssets>
 <us-gaap:AvailableForSaleSecuritiesDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">83875</us-gaap:AvailableForSaleSecuritiesDebtSecurities>
 <boa:AvailableforsaleSecuritiesDebtSecuritiesCollateral contextRef="ctx1" unitRef="USD" decimals="-6">40</boa:AvailableforsaleSecuritiesDebtSecuritiesCollateral>
 <us-gaap:HeldToMaturitySecurities contextRef="ctx1" unitRef="USD" decimals="-6">40</us-gaap:HeldToMaturitySecurities>
 <us-gaap:HeldToMaturitySecuritiesFairValue contextRef="ctx1" unitRef="USD" decimals="-6">192846</us-gaap:HeldToMaturitySecuritiesFairValue>
 <boa:TotalDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">706490</boa:TotalDebtSecurities>
 <us-gaap:LoansAndLeasesReceivableGrossCarryingAmount contextRef="ctx1" unitRef="USD" decimals="-6">9016</us-gaap:LoansAndLeasesReceivableGrossCarryingAmount>
 <us-gaap:LoansAndLeasesReceivableNetReportedAmount contextRef="ctx1" unitRef="USD" decimals="-6">697474</us-gaap:LoansAndLeasesReceivableNetReportedAmount>
 <us-gaap:PropertyPlantAndEquipmentNet contextRef="ctx1" unitRef="USD" decimals="-6">9255</us-gaap:PropertyPlantAndEquipmentNet>
 <boa:MortgageServicingRights contextRef="ctx1" unitRef="USD" decimals="-6">3045</boa:MortgageServicingRights>
 <us-gaap:ServicingAssetAtFairValueAmount contextRef="ctx1" unitRef="USD" decimals="-6">20</us-gaap:ServicingAssetAtFairValueAmount>
 <us-gaap:Goodwill contextRef="ctx1" unitRef="USD" decimals="-6">65662</us-gaap:Goodwill>
 <us-gaap:IntangibleAssetsNetExcludingGoodwill contextRef="ctx1" unitRef="USD" decimals="-6">9422</us-gaap:IntangibleAssetsNetExcludingGoodwill>
 <us-gaap:OtherAssets contextRef="ctx1" unitRef="USD" decimals="-6">119683</us-gaap:OtherAssets>
 <unit id="USD">
 <measure xmlns:iso4217="http://www.xbrl.org/2003/iso4217">iso4217:USD</measure>
 </unit>
</xbrl>
```

## 26.4 Beispiel Datenbank - XBRL

Dieses Beispiel steht unter dem Namen **DB\_to\_XBRL.mfd** im Ordner ...MapForceExamples zur Verfügung. Zum Extrahieren der Datenbankdaten werden darin verschiedene Filter und Funktionen verwendet.

Die Taxonomie **nanonull.xsd** wurde von US:GAAP abgeleitet. Im Mapping wird eine XBRL-Output-Instanz erstellt, die alle Contexts, Concepts, Units und Fußnoten für eine "Disclosure" und drei "Statements" (Berichte).

### Im Bericht "091 - Disclosure - Segment Revenue and Operating Income"

sehen Sie, wie Dimension-Werte in MapForce gemappt werden können. Der Hypercube "us-gaap:ScheduleOfSegmentReportingInformationBySegmentTable" enthält eine explizite Dimension "us-gaap:StatementOperatingActivitiesSegmentAxis".

Ihre Domain wurde in der Taxonomie durch drei Dimensionswerte, nämlich "nanonull:USA", "nanonull:Europe" und "nanonull:Asia" erweitert. Im Mapping sehen Sie, wie die Werte der Datenbankspalte "Name" der Tabelle "Region" mittels einer Wertezuordnungsfunktion (value-map) auf die erforderlichen Dimension-Werte vom Typ QName gemappt werden.

### Im Bericht "106 - Statement - Nanonull and Consolidated Statement of Cash Flows"

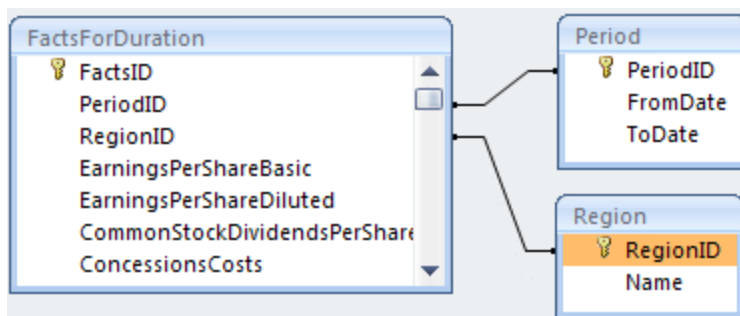
sehen Sie, wie mit Hilfe von MapForce Facts in die Output-Instanz, die sowohl mit Duration als auch Instant Periods verknüpft ist, geschrieben werden können.

Wie im Mapping gezeigt, können Facts wie z.B. "Cash and cash equivalents at beggining (end) of period" durch Duplizieren des Period Item in der Hierarchiestruktur ordnungsgemäß in einem Bericht ausgegeben werden.

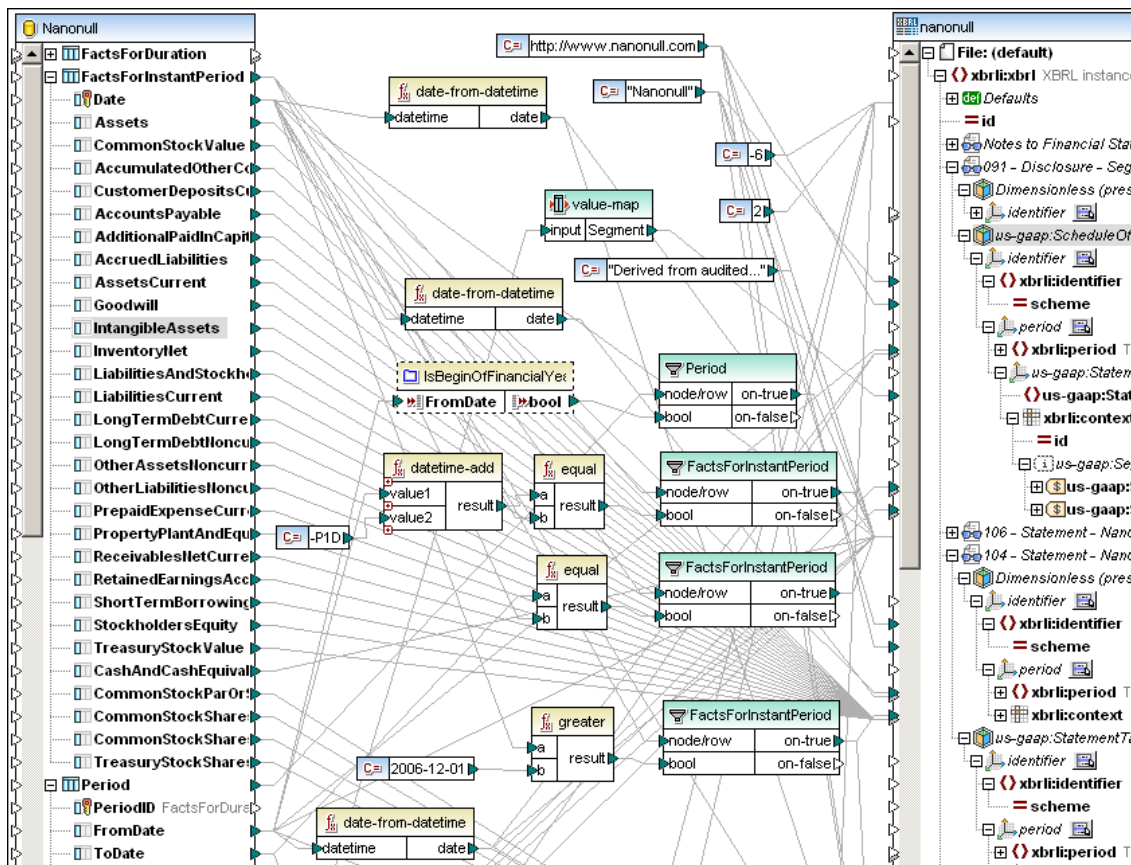
Im Mapping sind in der Output-Instanz zwei Units definiert: "USD" und "perShareItemtype". Das xbrli:unit Element muss zu diesem Zweck dupliziert werden. Die dazugehörigen measure-Elemente werden mit Hilfe der Funktionen "xbrl-measure-currency" und "xbrl-measure-shares" anhand der XBRL-Bibliothek erstellt.

Die Facts in den Datenbanktabellen wurden aufgeteilt, je nachdem, ob sie sich auf eine Instant oder Duration Period beziehen.

- Die Tabelle **FactsForInstantPeriod** ist eine flache (nicht hierarchische) Wertetabelle.
- Die Tabelle **FactsForDuration** ist hierarchisch und jedes Fact, das darin enthalten ist, bezieht sich auf eine bestimmte PeriodID sowie eine RegionID.



In der Period-Tabelle werden die Daten der Start und End Period anhand der Felder FromDate und ToDate definiert, während sich in der Region-Tabelle jedes der Facts auf eine bestimmte Region bezieht, also Asia, Europe oder USA.



Obligatorische XBRL Items, die in einer XBRL-Instanzdatei benötigt werden:

- **unitRef** und entweder **decimals** oder **precision** in Monetary Concepts.
- **xbrli:identifizier** und **scheme** der Identifier Dimension
- **xbrl:period** und entweder **xbrli:instant** oder die Elemente **xbrli:startDate/xbrli:endDate**
- **xbrli:id** und **xbrli:measure** im Element **xbrli:unit**

Unter [Obligatorische Elemente - Standardeinheiten](#) sehen Sie, wie dies in einem anderen Beispiel durchgeführt wurde.

# Kapitel 27

---

Unterstützung für QName

## 27 Unterstützung für QName

Mit Hilfe von QNames (qualifizierten Namen) können Sie in XML- und XBRL-Instanzdokumenten verwendete Namespace URIs referenzieren und abkürzen. Es gibt zwei Arten von QNames; Namen mit und ohne Präfix.

PrefixName    Präfix ': LocalPart

UnPrefixName    LocalPart

wobei LocalPart ein Element- oder Attributname ist.

```
<Doc xmlns:x="http://myCompany.com">
 <x:part>
</Doc>
```

x ist die Namespace-Referenz auf "**http://myCompany.com**" und <x:part> ist daher ein gültiger QName, da:

**x** das Namespace-Präfix ist und  
**part** der LocalPart ist, d.h. der Elementname.

MapForce unterstützt die folgenden QName-Funktionen im Abschnitt **Lang** des Bibliotheksfensters:

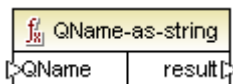
### QName

Konstruiert anhand einer Namespace URI und eines lokalen Teils einen QName. Mit Hilfe dieser Funktion können Sie einen QName in einer Zielkomponente erstellen. Der uri- und localname-Parameter können durch eine Konstantenfunktion bereitgestellt werden.



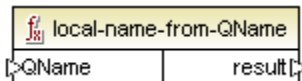
### QName-as-string

Konvertiert einen QName in einen String in der Form **{http://myCompany.com}local**.



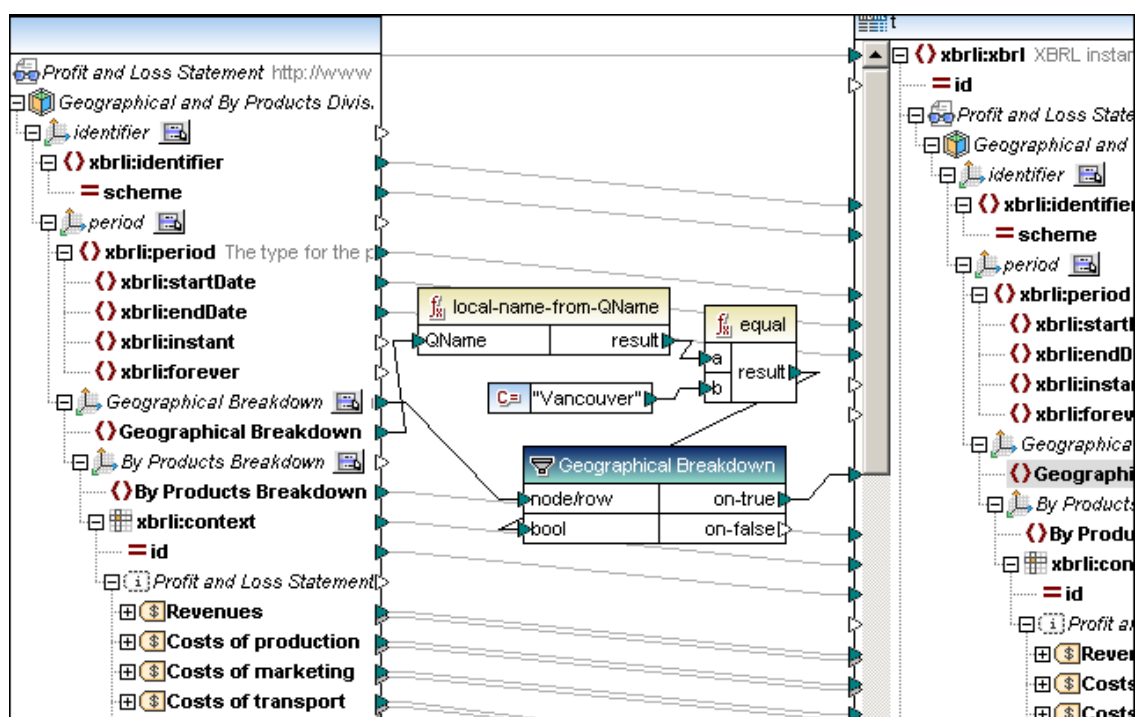
### local-name-from-QName

Extrahiert aus einem QName den lokalen Namen.



Diese Funktion ist beim Mappen von XBRL-Instanzdokumenten, die Hypercubes enthalten extrem nützlich.





Beim Mapping werden jene Facts, bei denen der **lokale Name** des **Inhalts** des explicit Member (d-g:Vancouver) gleich "Vancouver" ist, herausgefiltert. Beachten Sie, dass der Inhalt des Member selbst ein QName ist.

```
<xbrli:context id="Year2007_Vancouver_BeveragesTotal">
 <xbrli:entity>
 <xbrli:identifier scheme="http://www.stockexchange/ticker">ACME</xbrli:identifier>
 <xbrli:segment>
 <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-g:Vancouver</xbrldi:explicitMember>
 <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-b:BeveragesTotal</xbrldi:explicitMember>
 </xbrli:segment>
 </xbrli:entity>
</xbrli:context>
```

Alle Facts, die zur Dimension "GeographicalBreakdown" gehören, werden gefiltert und an die Zielkomponente übergeben.

```
<context id="Year2007_Vancouver_BeveragesTotal">
 <entity>
 <identifier scheme="http://www.stockexchange/ticker">ACME</identifier>
 <segment>
 <explicitMember dimension="d-b:ByProductsBreakdown" xmlns="http://xbrl.org/2006/xbrldi">d-b:BeveragesTotal</explicitMember>
 <explicitMember dimension="d-g:GeographicalBreakdown" xmlns="http://xbrl.org/2006/xbrldi">d-g:Vancouver</explicitMember>
 </segment>
 </entity>
 <period>
 <startDate>2007-01-01</startDate>
 <endDate>2007-12-31</endDate>
 </period>
</context>
<p:Revenues contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">4</p:Revenues>
<p:CostsOfProduction contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">1</p:CostsOfProduction>
<p:CostsOfMarketing contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">1</p:CostsOfMarketing>
<p:ProfitLoss contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">2</p:ProfitLoss>
```

**namespace-uri-from-QName**  
Extrahiert aus einem QName die Namespace URI.

|                                          |        |
|------------------------------------------|--------|
| f <sub>NS</sub> namespace-uri-from-QName |        |
| QName                                    | result |

# Kapitel 28

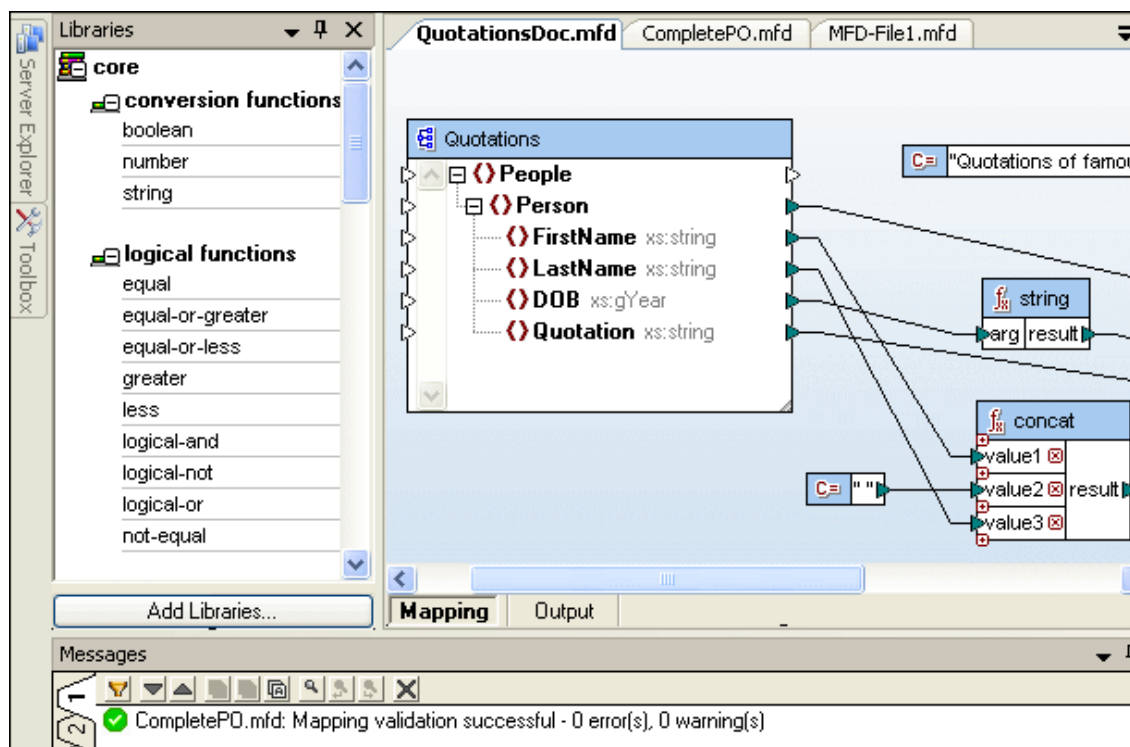
---

MapForce Plug-in für MS Visual Studio

## 28 MapForce Plug-in für MS Visual Studio

Ihre Version von MapForce2012 lässt sich in die Microsoft Visual Studio IDE-Versionen 2005, 2008 und 2010 integrieren. Dadurch stehen Ihnen die Vorteile beider Umgebungen, nämlich komplexe Mapping-Funktionen in der umfangreichen Entwicklungsumgebung Visual Studio gleichzeitig zur Verfügung. Sie benötigen dazu die folgenden Voraussetzungen:

- Installieren Sie Microsoft Visual Studio
- Installieren Sie MapForce (Enterprise oder Professional Edition)
- Laden Sie das **MapForce Integrationspaket** herunter und führen Sie es aus. Dieses Paket steht auf der MapForce (Enterprise und Professional Edition) Download-Seite unter [www.altova.com](http://www.altova.com) zur Verfügung



Nach Installation des Integrationspakets können Sie MapForce in der Visual Studio-Umgebung verwenden.

### Aktivierung des Plug-In

Möglicherweise wurde das Plug-In während der Installation nicht aktiviert.

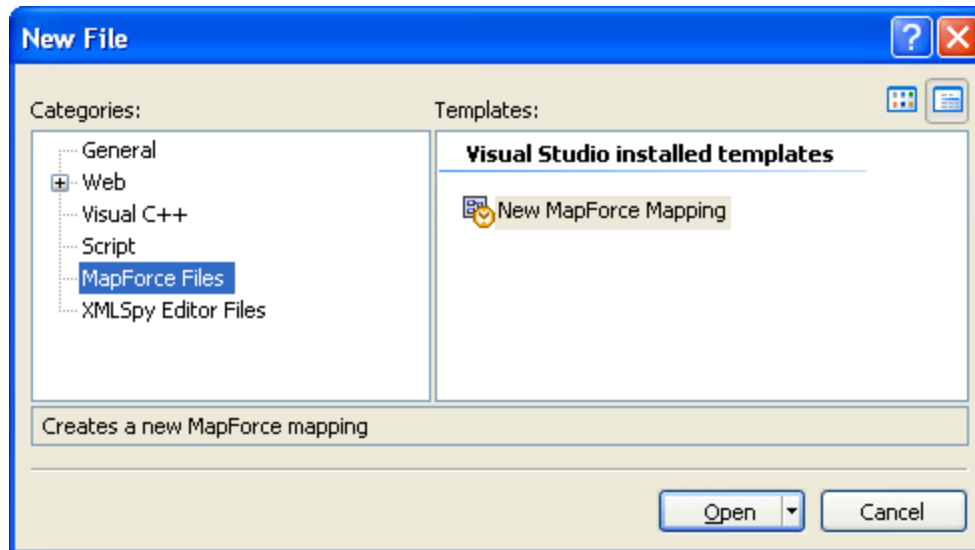
#### So aktivieren Sie das Plug-In:

1. Gehen Sie zum **Verzeichnis**, in dem die ausführbare Visual Studio IDE Datei installiert wurde, z.B. `c:\Program Files\Microsoft Visual Studio 8\Common7\IDE`
2. Geben Sie in die Befehlszeile den folgenden Befehl ein: **devenv.exe /setup**.
3. Warten Sie bis der Vorgang abgeschlossen ist, bevor Sie die Applikation in Visual Studio öffnen.

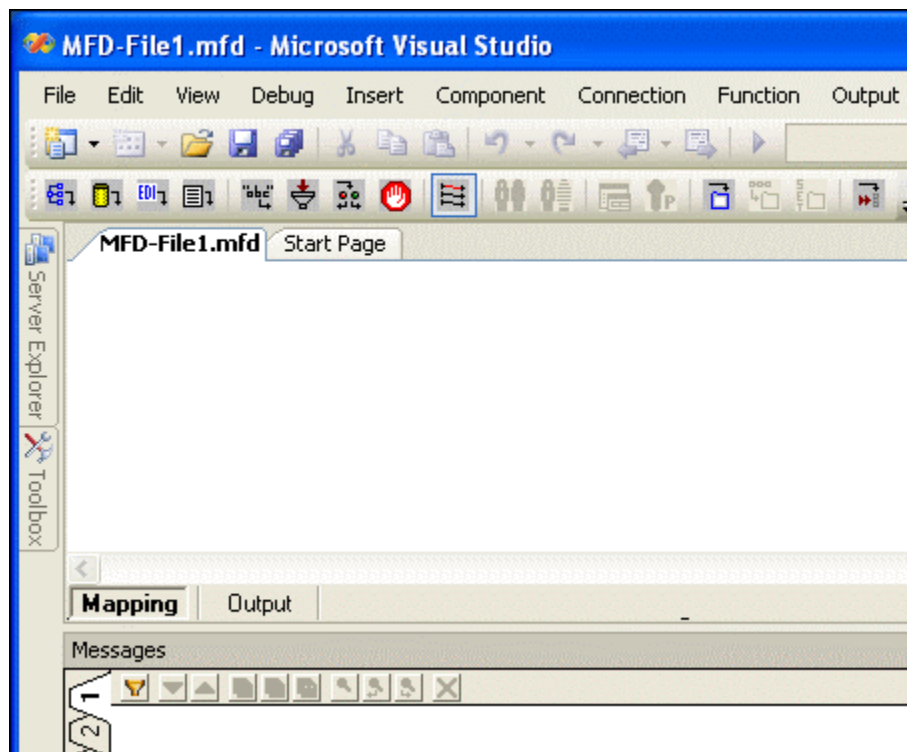
## 28.1 Öffnen von MapForce Dateien in Visual Studio

So öffnen Sie eine neue MapForce Mapping-Datei:

1. Wählen Sie die Menüoption **Datei | Neu**.
2. Klicken Sie unter Kategorien auf den Eintrag MapForce Files.

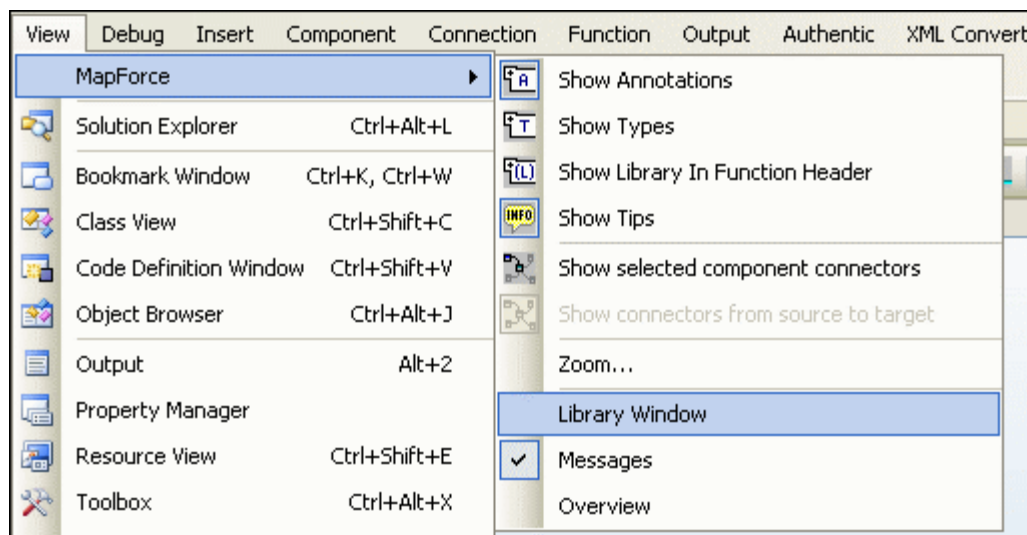


3. Doppelklicken Sie im Fenster Templates auf den Eintrag "New MapForce Mapping". Daraufhin wird eine leere Mapping-Datei geöffnet.



So aktivieren Sie das Fenster "Bibliotheken":

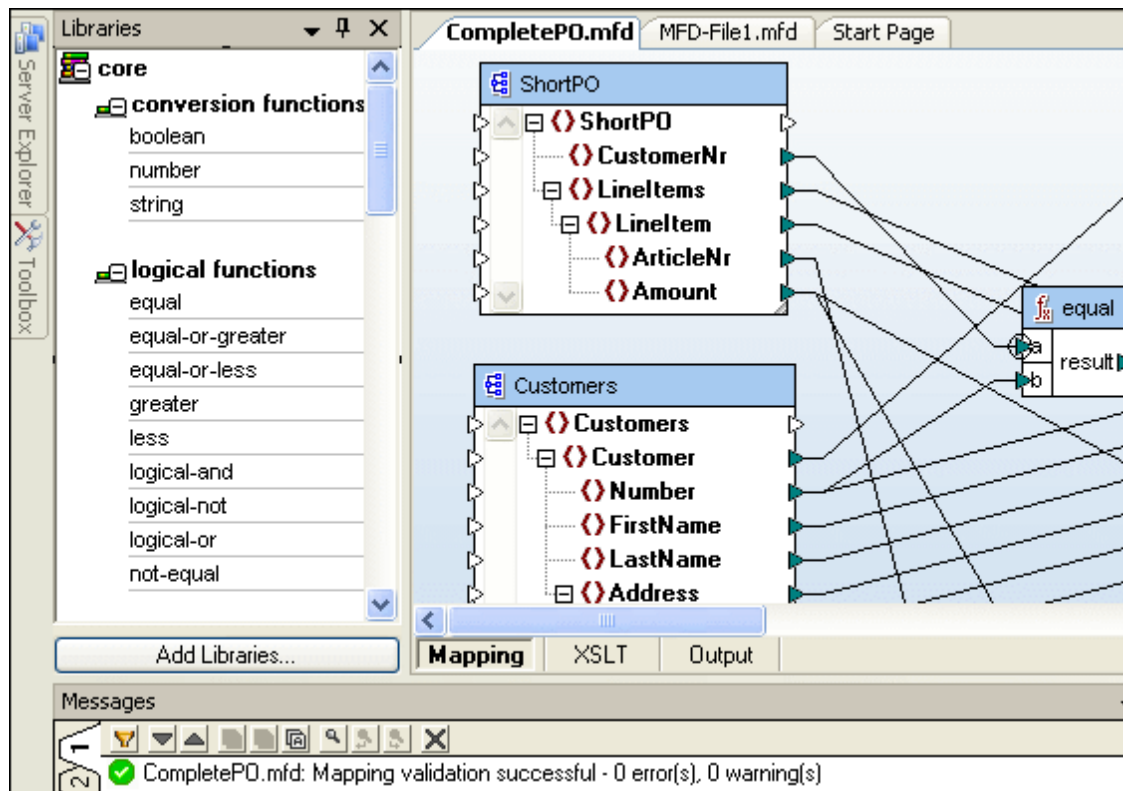
1. Wählen Sie den Menüeintrag **Ansicht | MapForce | Bibliotheksfenster**.



2. Docken Sie das freischwebende Fenster an der gewünschten Position an, z.B. am linken Rand.

#### So öffnen Sie eine mitgelieferte Beispieldatei:

1. Wählen Sie die Menüoption **Datei | Öffnen**, navigieren Sie zum Ordner **...MapForceExamples** und öffnen Sie eine MapForce-Datei. Im Screenshot unten sehen Sie die Datei **CompletePO.mfd**.



## 28.2 Unterschiede zwischen Visual Studio und Standalone-Versionen

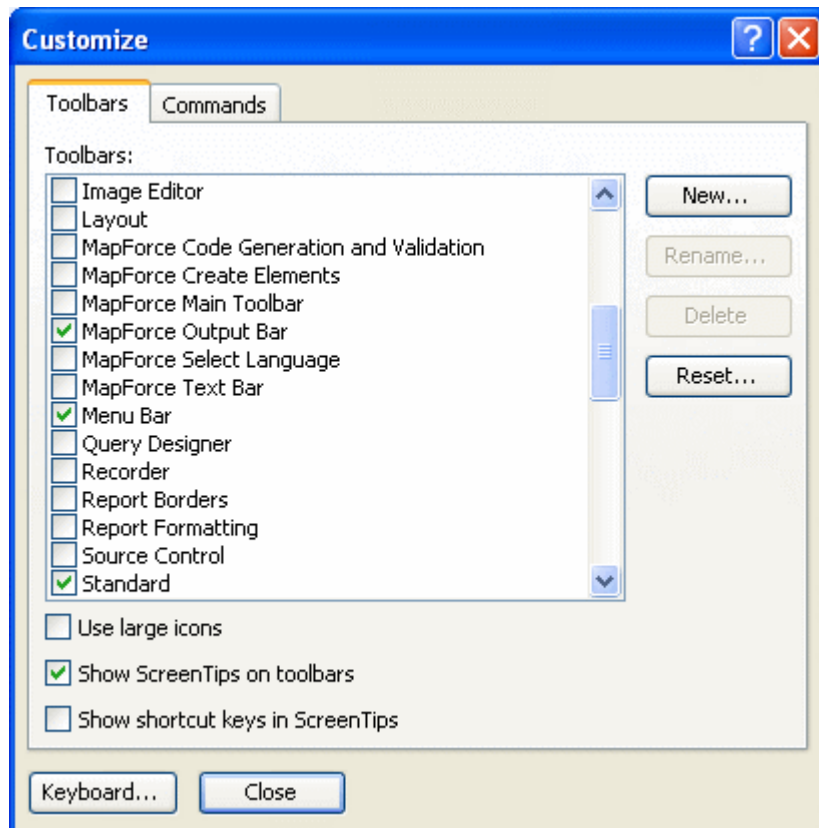
Die MapForce Enterprise und Professional-Plug-ins sind alle auf die gleiche Art in alle Versionen von MS Visual Studio integriert. Beachten Sie bitte, dass MapForce Projekte in der Visual Studio Version nicht unterstützt werden.

### Geänderte Funktionen in den Visual Studio Versionen:

Menü **Bearbeiten**, Rückgängig und Wiederherstellen

Die Befehle "Rückgängig" und "Wiederherstellen" wirken sich auf alle Aktionen aus (Kopieren Einfügen usw.), die in der Entwicklungsumgebung ausgeführt wurden, auch auf alle Aktionen in MapForce.

Menü **Extras | Anpassen | Symbolleiste, Befehle.**



Diese Register enthalten sowohl Visual Studio- als auch MapForce-Befehle.

Menü **Ansicht**

Das Menü "Ansicht" enthält das Untermenü MapForce, über das Sie MapForce Tool-Fenster aktivieren und deaktivieren können. Sie haben über dieses Menü auch Zugriff auf die MapForce Ansichtseinstellungen.

Menü **Hilfe**

Das Menü **Hilfe** enthält das Untermenü MapForce Hilfe, über das Sie die MapForce Hilfe aufrufen können. Außerdem enthält dieses Menü Links zum Altova Support Center, Komponenten-Download-Bereich usw.

### Nicht unterstützte Funktionen der Visual Studio-Version von MapForce

Weder das **Projektfenster** noch das Menü **Projekte** stehen in diesen Versionen zur Verfügung,

d.h. MapForce-Projekte und WSDL-Projekte können in diesen Versionen nicht geöffnet werden.



# Kapitel 29

---

## MapForce Plug-in für Eclipse

## 29 MapForce Plug-in für Eclipse

Eclipse ist ein Open Source Framework, in das unterschiedliche Arten von Applikationen in Form von Plug-ins installiert werden. MapForce für die Eclipse-Plattform ist ein Eclipse Plug-in, über das Sie von der Eclipse 3.5-Plattform (oder höher) Zugriff auf die Funktionalitäten einer zuvor installierten MapForce Version haben.

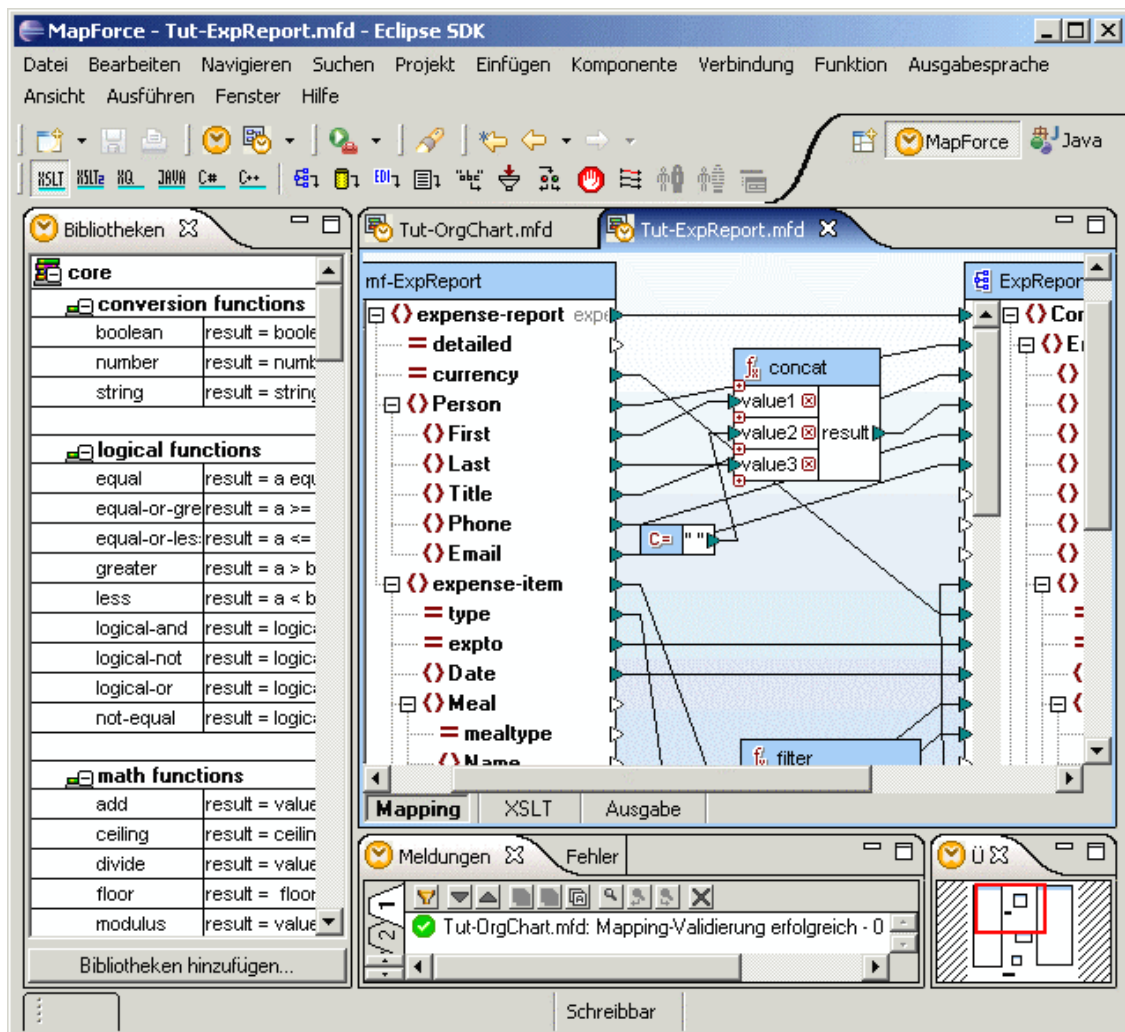
### Installationsanforderungen

Um das MapForce Plug-in für Eclipse 3.5 oder höher installieren zu können, benötigen Sie die folgenden Komponenten:

- die jeweilige MapForce Version, die Sie verwenden möchten: Enterprise oder Professional
- das Eclipse-Paket sowie
- die entsprechende Java Runtime Edition
- Downloaden Sie das **MapForce Integrationspakete für Eclipse** und führen Sie es aus. Dieses Paket steht auf der MapForce Download-Seite unter [www.altova.com](http://www.altova.com) zur Verfügung.

Das MapForce Plug-in für Eclipse bietet die folgenden Funktionalitäten:

- Ein Daten-Mapping-Tool mit allen Funktionen zur Durchführung komplexer Datenintegrationsprojekte.
- Codegenerierungsfunktionen in den von der jeweiligen Version unterstützten Programmiersprachen.
- MapForce Benutzerhilfe unter dem Menüeintrag **Hilfe | MapForce| Inhaltsverzeichnis**.



### Voraussetzungen für die Java Run-Time-Umgebung (JavaRTE):

Das Eclipse Plug-In unterstützt die Eclipse-Versionen 3.5 und höher, für die eine JavaRTE (Run-Time Environment) der Version 1.5 oder höher benötigt wird.

Wenn beim Öffnen eines Dokuments die unten stehenden Fehlermeldung angezeigt wird, weist dies darauf hin, dass Eclipse eine ältere JavaRTE verwendet. Eclipse verwendet zum Aufrufen der Datei `javaw.exe` die Umgebungsvariable `PATH`.

Fehler:

**java.lang.UnsupportedClassVersionError: com/altova/.... (Unsupported major.minor version 49.0)**

Das Problem kann folgendermaßen gelöst werden:

- Führen Sie Eclipse entweder mit dem Befehlszeilenparameter **-vm** und unter Angabe des Pfads zu einer **javaw.exe** der Version 1.5 oder höher aus.
- oder überprüfen Sie die Variable `PATH` zum Auffinden der Datei `javaw.exe`, die als erste gefunden wird, (wenn mehrere Installationen von Eclipse vorhanden sind) und ändern Sie diese so, dass Sie auf die neuere Version verweist.

## 29.1 Manuelle Installation des MapForce Plug-in

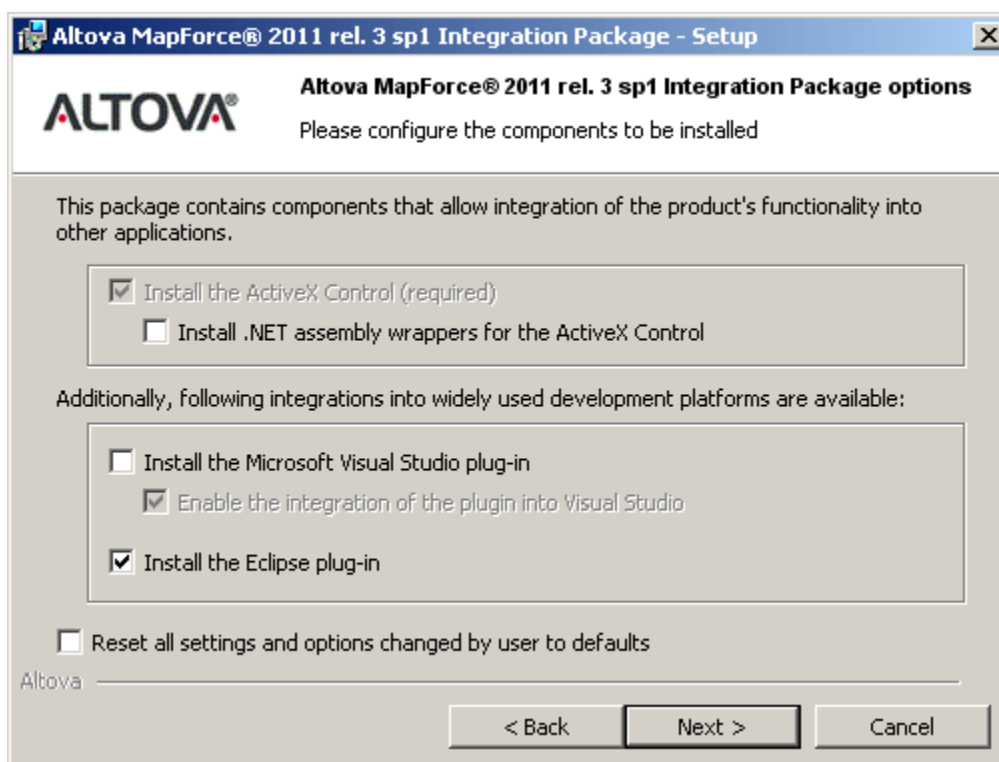
### Manuelle Installation des MapForce Plug-in für Eclipse:

So installieren Sie das MapForce Plug-in:

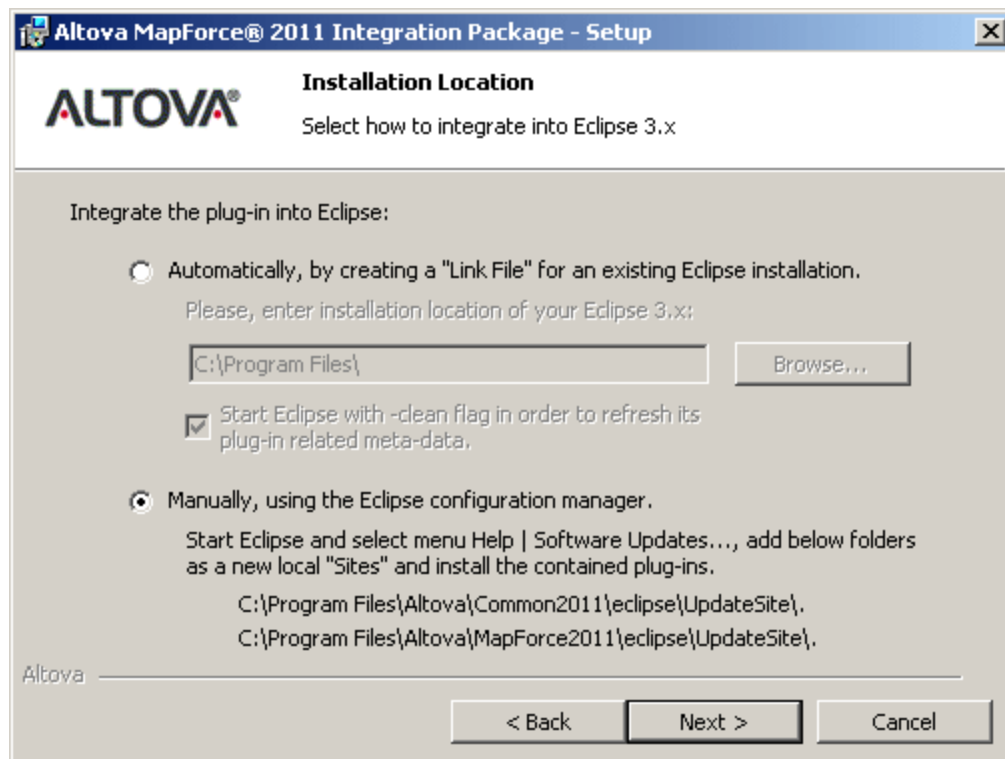
- Laden Sie die gewünschte MapForce Version (also die Enterprise, Professional oder Basic Edition) vom Download-Bereich der Altova.com Website herunter und installieren Sie sie.
- Laden Sie das MapForce Integration Package vom Download-Bereich [Komponenten: IDE-Integration](#) der Altova.com Website herunter und installieren Sie es. Sie werden während der Installation aufgefordert, den Installationsordner von Eclipse anzugeben.

Nachdem Sie sich mit den Lizenzbedingungen einverstanden erklärt haben und auf "Weiter" geklickt haben,

1. Aktivieren Sie die Option "Eclipse Plug-In installieren" und klicken Sie auf "Weiter".



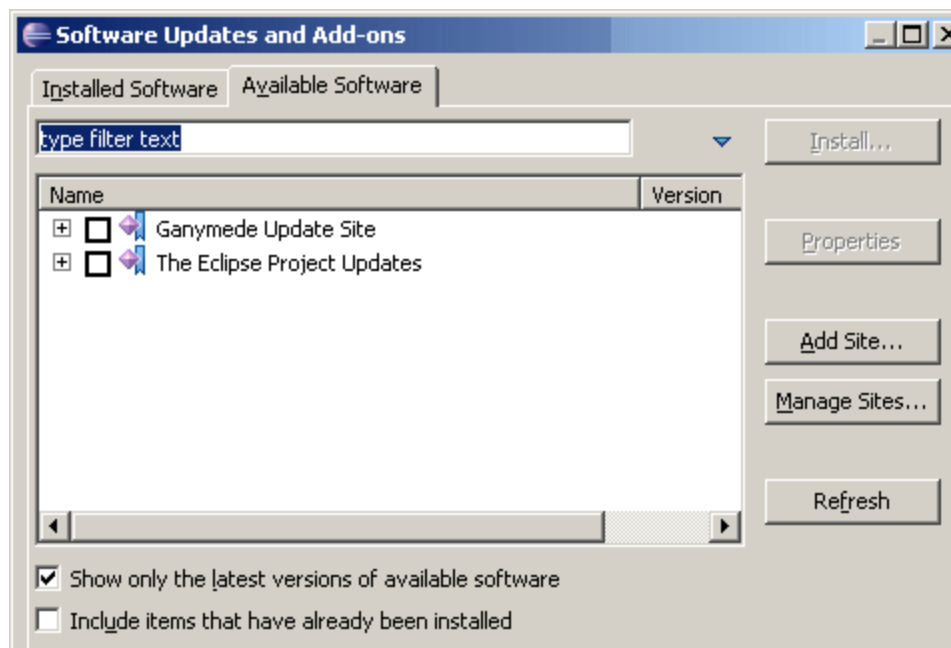
2. Klicken Sie auf "manuell über den Eclipse Konfigurationsmanager", klicken Sie anschließend auf "Weiter" um fortzufahren.



3. Klicken Sie auf "Installieren" und anschließend auf "Fertig stellen", um die Installation abzuschließen.

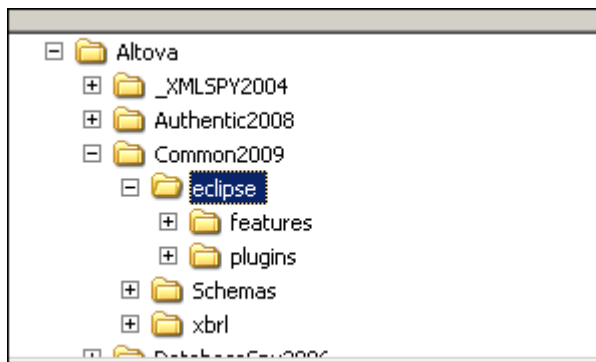
Konfigurieren der Eclipse-Installation für die Verwendung eines **zuvor** installierten MapForce Plug-in:

1. Starten Sie Eclipse und wählen Sie die Menüoption **Hilfe | Software Updates**.

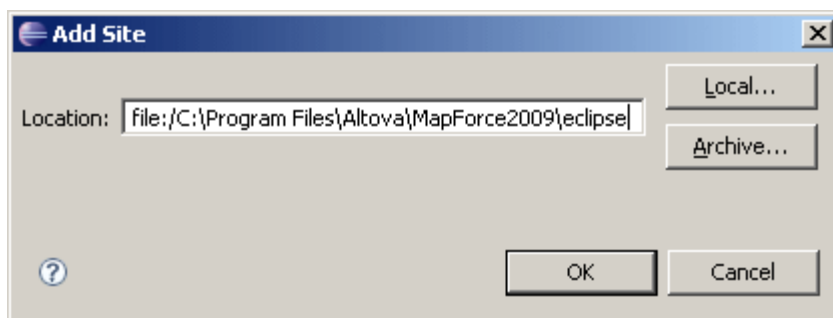


2. Klicken Sie auf die Schaltfläche **Add Site** und anschließend auf die Schaltfläche "Local"
3. Navigieren Sie zum Ordner ...**Program Files\Altova\Common2012**, wählen Sie den

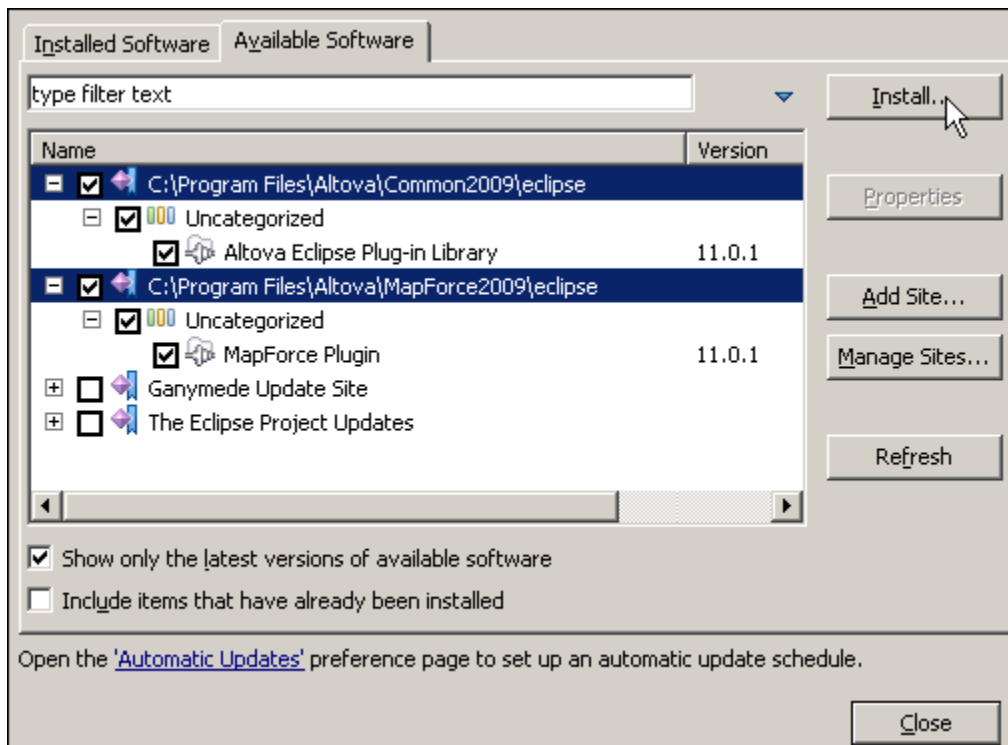
Ordner "Eclipse" und klicken Sie zum Fortfahren auf Weiter.



Klicken Sie nochmals auf die Schaltfläche "Add Site", anschließend auf "Local" und navigieren Sie zu **...\\Program Files\\Altova\\MapForce2012\\eclipse** und klicken Sie auf OK, um fortzufahren.

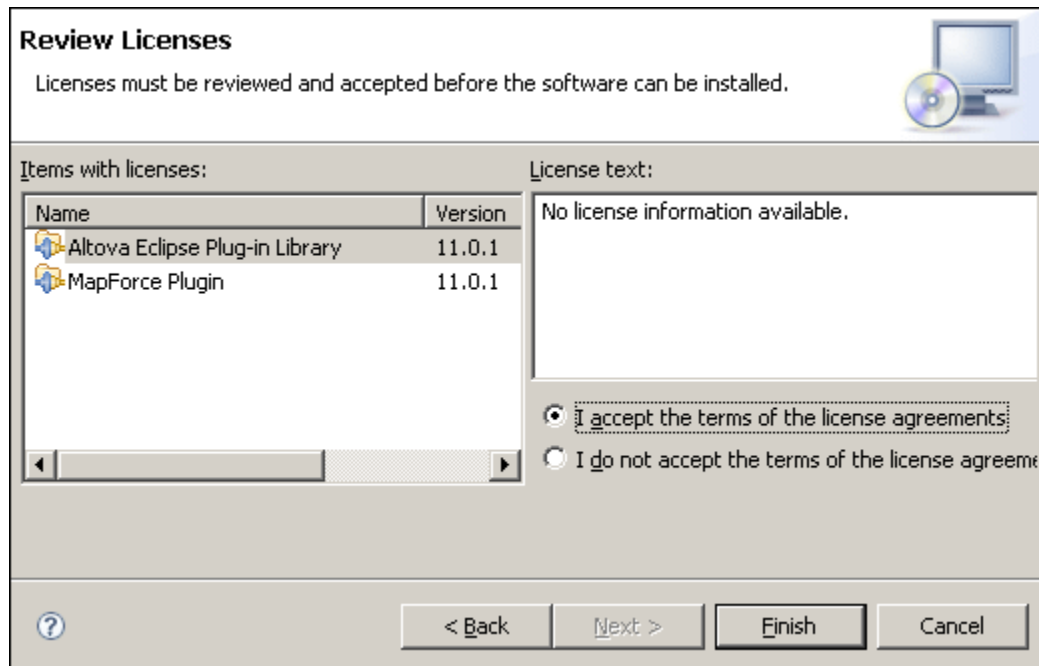


5. Aktivieren Sie die Kästchen auf der obersten Ebene der einzelnen Pfadeinträge, um die Plug-ins auszuwählen und klicken Sie auf **Install** um fortzufahren.



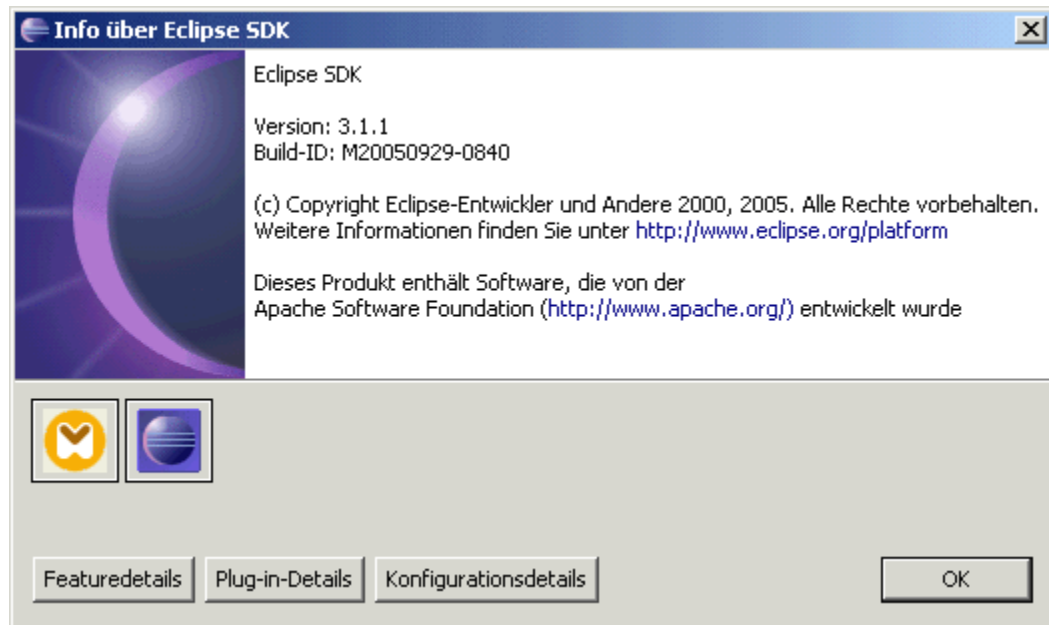
Daraufhin wird ein Dialogfeld geöffnet, in dem Sie überprüfen können, ob die angeklickten Komponenten installiert werden.

6. Klicken Sie zum Fortfahren auf **Next**.
7. Daraufhin wird das Dialogfeld "Review License" angezeigt. Klicken Sie auf *I accept the terms....*, wenn Sie damit einverstanden sind. Klicken Sie anschließend auf **Finish**, um die Installation fertig zu stellen.



**So überprüfen Sie, welche Version derzeit installiert ist:**

1. Wählen Sie die Menüoption **Hilfe | Info über Eclipse SDK**.



2. Klicken Sie auf das MapForce Symbol, um die Informationen zur Version anzuzeigen.

**Anmerkung:**

Wenn das Plug-in nicht als installiert aufscheint, oder Symbolleisten-Schaltfläche fehlen, starten Sie Eclipse einmal mit dem "clean" Parameter.

z.B. **eclipse.exe -clean**.

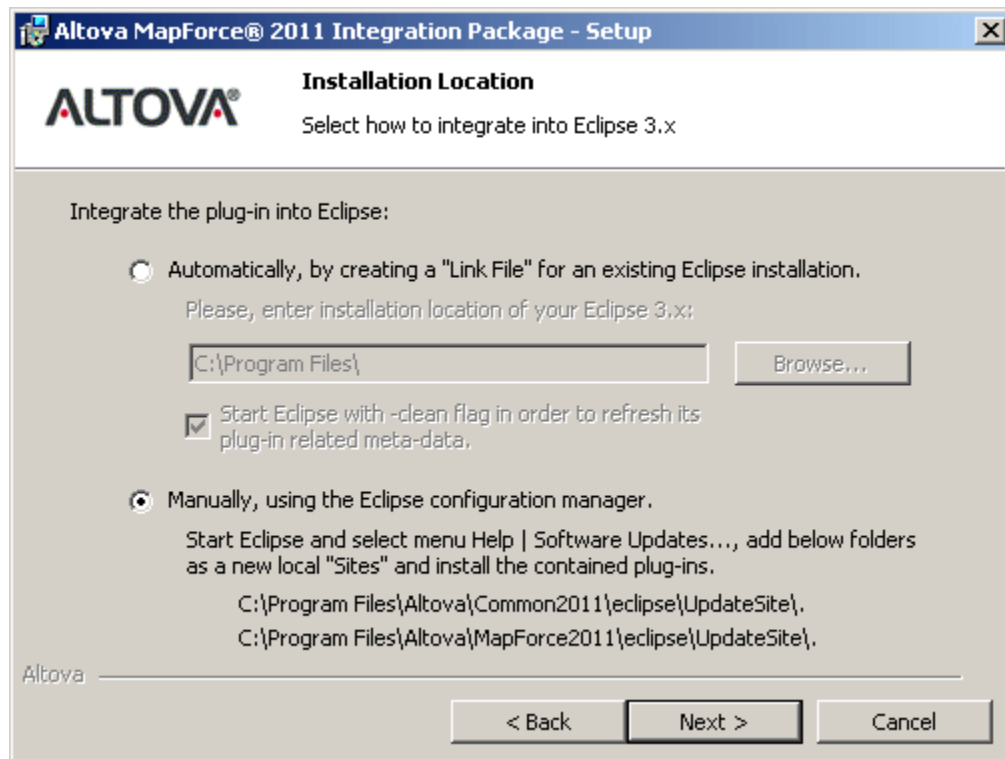


## 29.2 Manuelle Installation des MapForce Plug-in - Eclipse 3.6

### Manuelle Installation des MapForce Plug-in für Eclipse:

So installieren Sie das MapForce Plug-in:

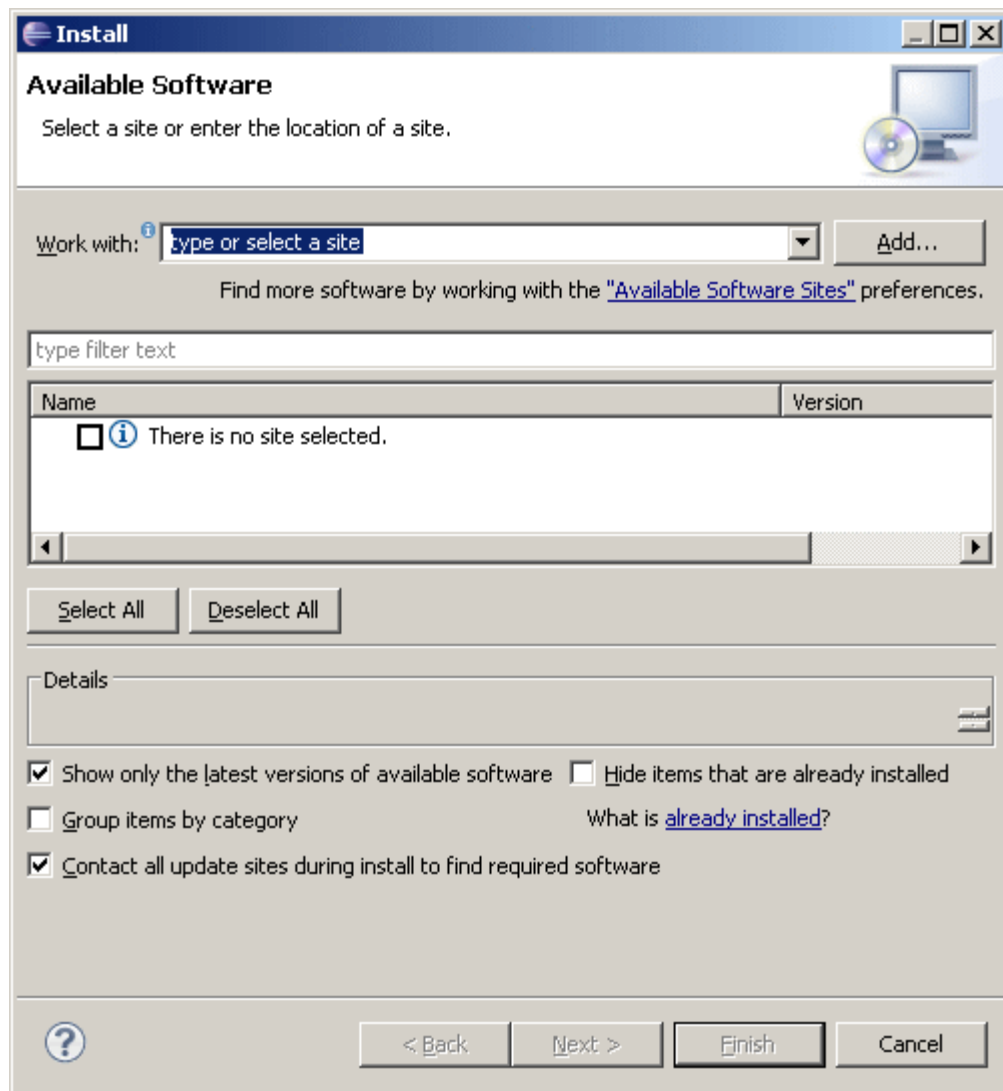
- Laden Sie die gewünschte MapForce Version (also die Enterprise, Professional oder Basic Edition) vom Download-Bereich der Altova.com Website herunter und installieren Sie sie.
- Laden Sie das MapForce Plug-in für Eclipse vom Download-Bereich der Altova.com Website herunter und installieren Sie es. Sie werden während der Installation aufgefordert, den Installationsordner von Eclipse anzugeben.



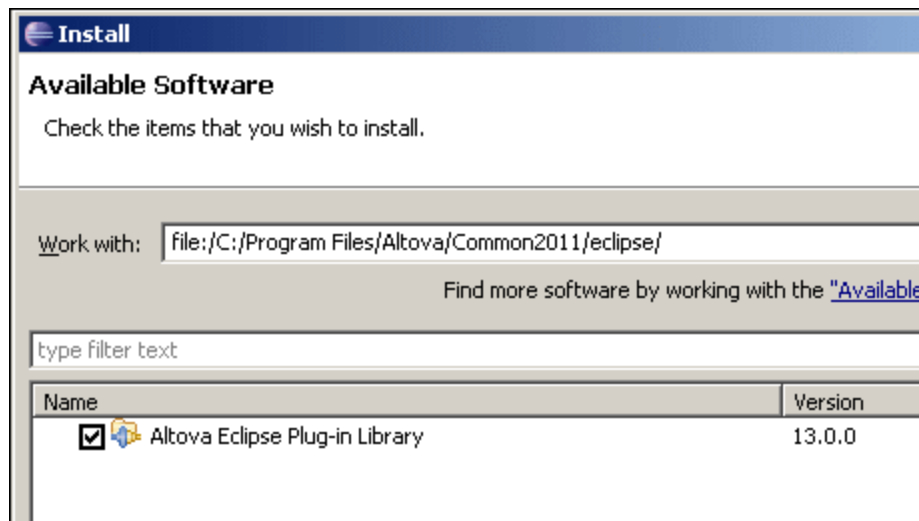
1. Klicken Sie auf "manuell über den Eclipse Konfigurationsmanager", klicken Sie anschließend auf "Weiter" um fortzufahren.
2. Klicken Sie auf "Installieren" und anschließend auf "Fertig stellen", um die Installation abzuschließen.

Konfigurieren der Eclipse-Installation für die Verwendung eines **zuvor** installierten MapForce Plug-in:

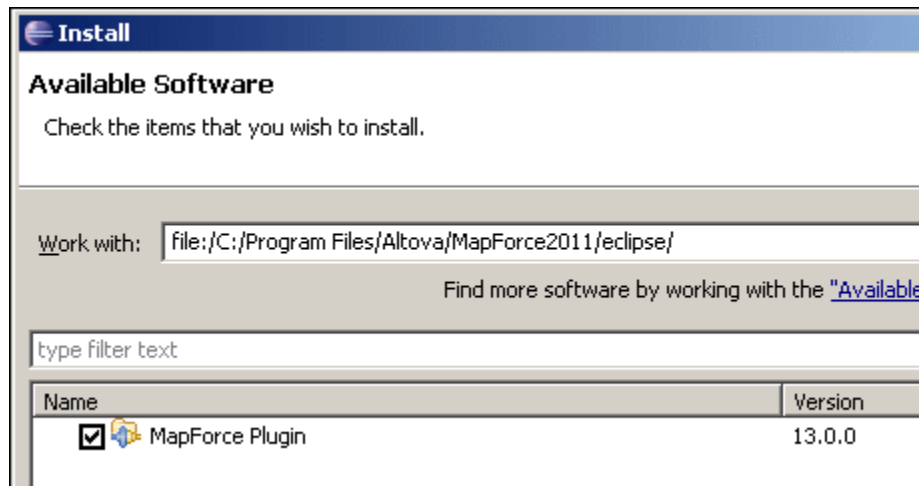
1. Starten Sie Eclipse und wählen Sie die Menüoption **Help | Install new Software**.



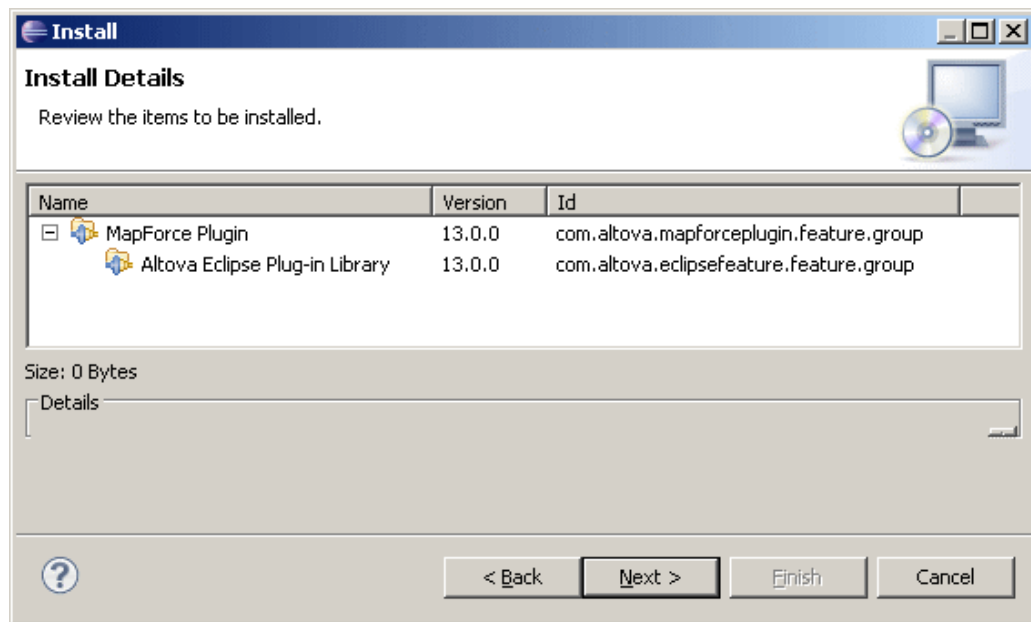
2. Klicken Sie auf die Schaltfläche **Add...** und anschließend auf die Schaltfläche "Local"
3. Navigieren Sie zum Ordner ...**Program Files\Altova\Common2012**, wählen Sie den Ordner "Eclipse" und klicken Sie zum Fortfahren auf OK.



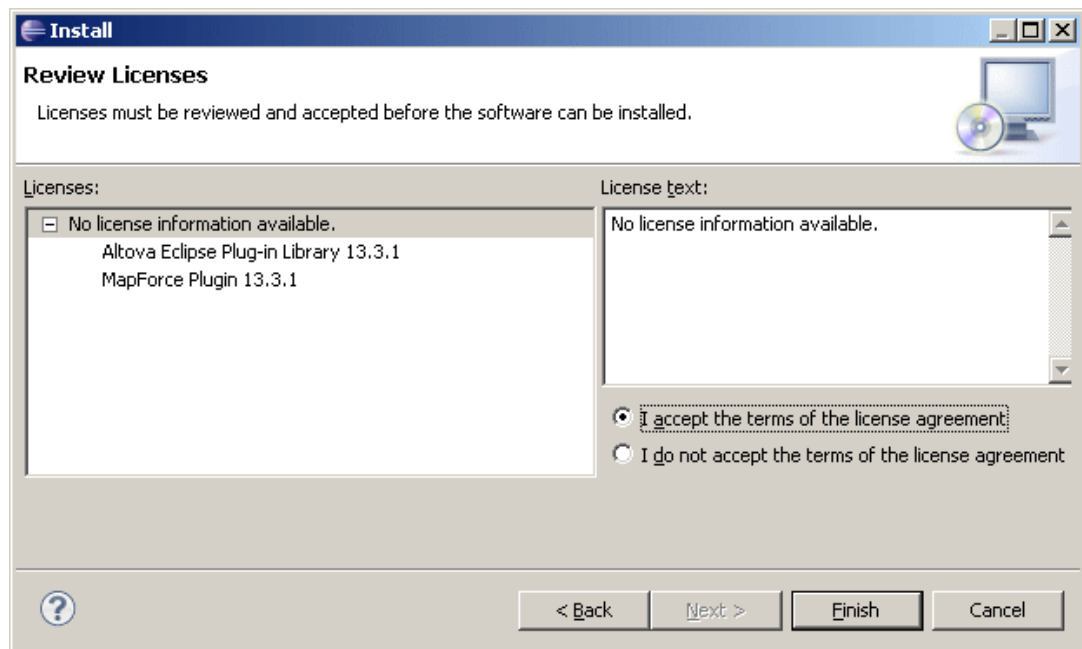
4. Klicken Sie auf das Kontrollkästchen "Altova Eclipse Plug-in Library", um es zu aktivieren.
5. Klicken Sie nochmals auf die Schaltfläche "Add...", anschließend auf "Local" und navigieren Sie zu **...\\...Program Files\\Altova\\MapForce2012\\eclipse** und klicken Sie auf OK, um fortzufahren.



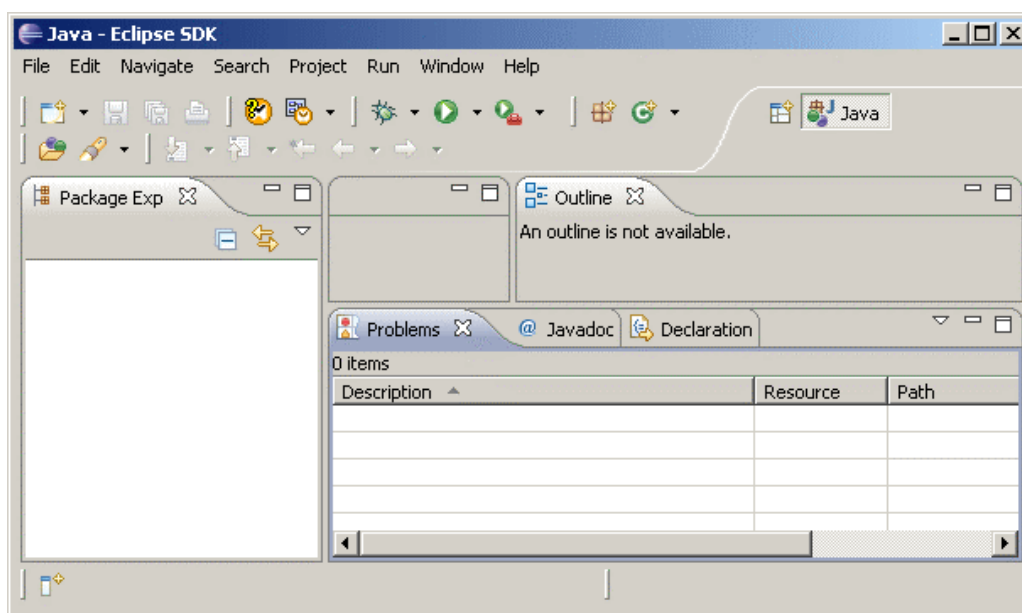
5. Klicken Sie auf das Kontrollkästchen "MapForce Plugin", um es auszuwählen.
6. Klicken Sie zum Fortfahren auf "Next...".  
Ein Dialogfeld zum Überprüfen der Installation wird geöffnet, in dem Sie die Installation der ausgewählten Einträge bestätigen können.



6. Klicken Sie zum Fortfahren auf "Next...".
7. Daraufhin wird das Dialogfeld zum Überprüfen der Lizenzen angezeigt. Aktivieren Sie das Optionsfeld "I accept the terms..." und klicken Sie auf "Fertig stellen" um die Installation fertig zu stellen.



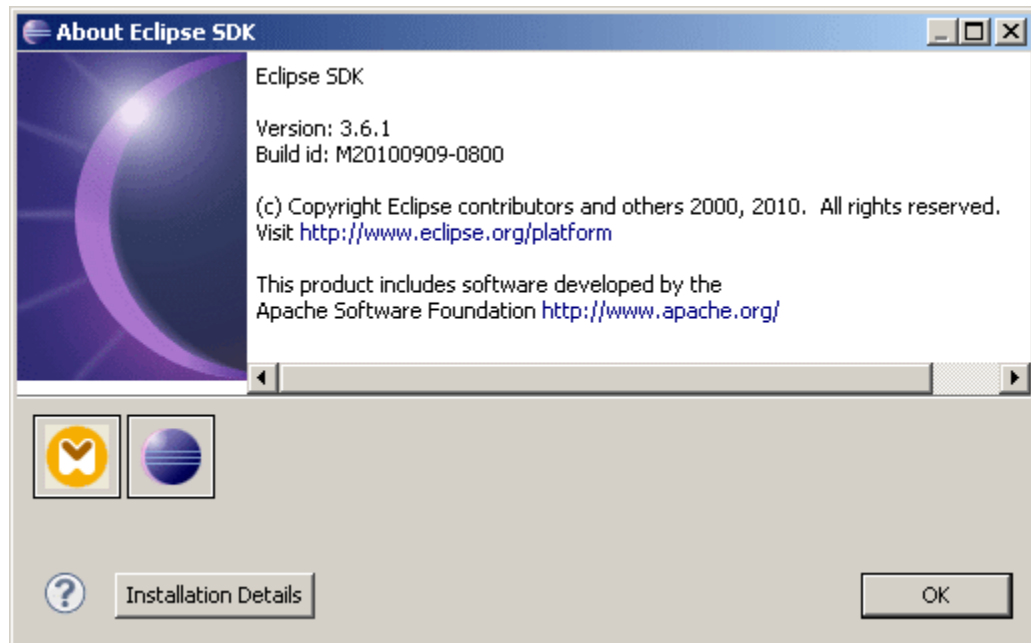
7. Es erscheint ein Dialogfeld mit der Warnung "Unsigned content...". Klicken Sie OK, damit das Plug-in installiert wird.
8. Klicken Sie auf "Restart eclipse", damit die Änderungen wirksam werden.



In der Hauptsymbolleiste werden nun zwei MapForce-Schaltflächen angezeigt.

**So überprüfen Sie, welche Version derzeit installiert ist:**

1. Wählen Sie die Menüoption **Hilfe | Info über Eclipse SDK**.



2. Klicken Sie auf die MapForce-Schaltfläche, um die Informationen zur Version anzuzeigen.

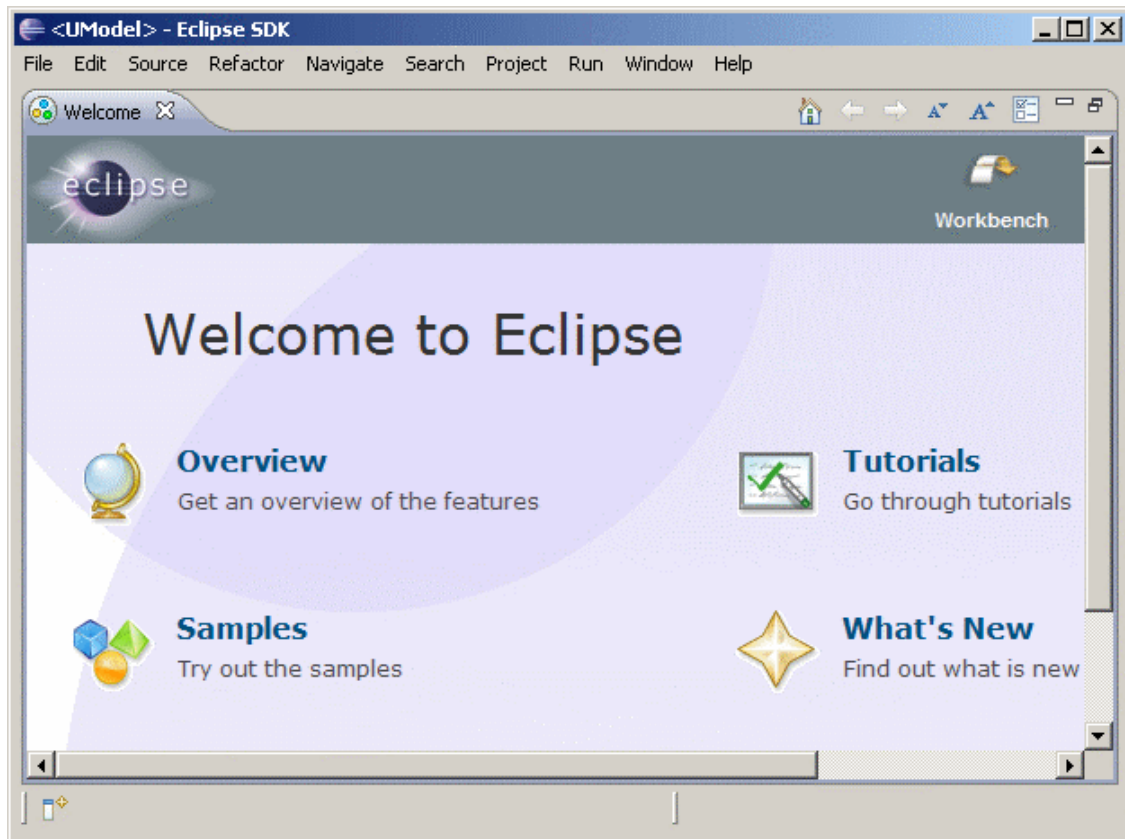
**Anmerkung:**

Wenn das Plug-in nicht als installiert aufscheint, oder Symbolleisten-Schaltfläche fehlen, starten Sie Eclipse einmal mit dem "clean" Parameter.

z.B. **eclipse.exe -clean**.

## 29.3 Starten von Eclipse und Verwendung des MapForce Plug-in

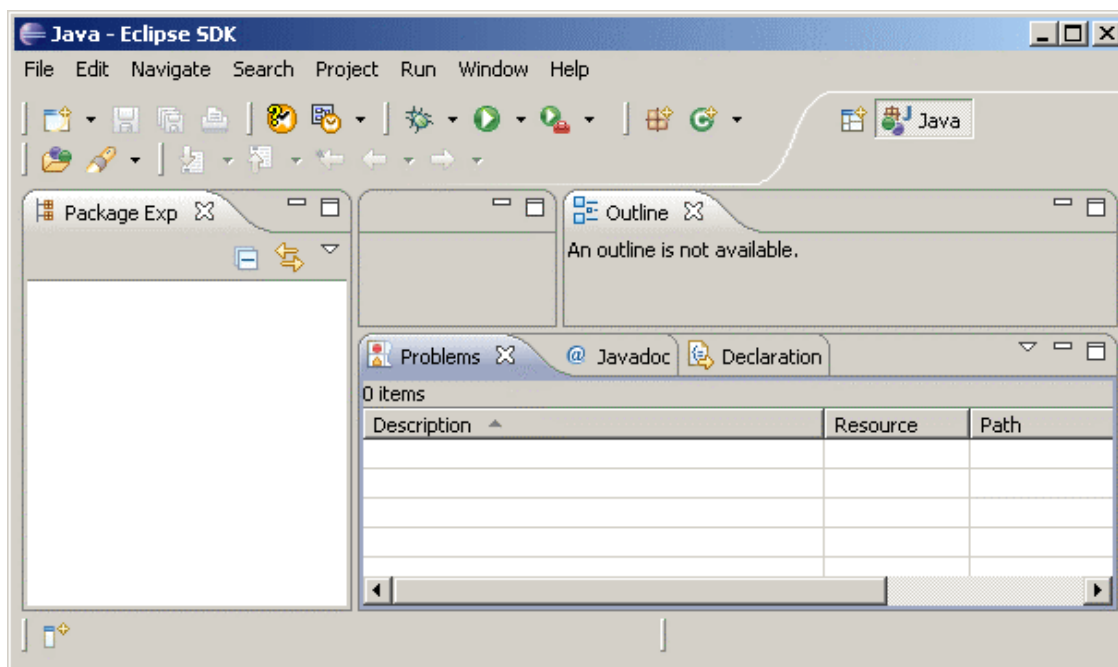
Platzieren Sie den Cursor über das Pfeilsymbol (rechts oben) und klicken Sie, um die "Workbench" zu öffnen. Daraufhin wird ein leeres MapForce Fenster in Eclipse geöffnet.



Wenn Sie diesen Begrüßungsbildschirm nicht sehen, können Sie jederzeit in der Eclipse IDE auf **Help | Welcome** klicken, um ihn anzuzeigen.

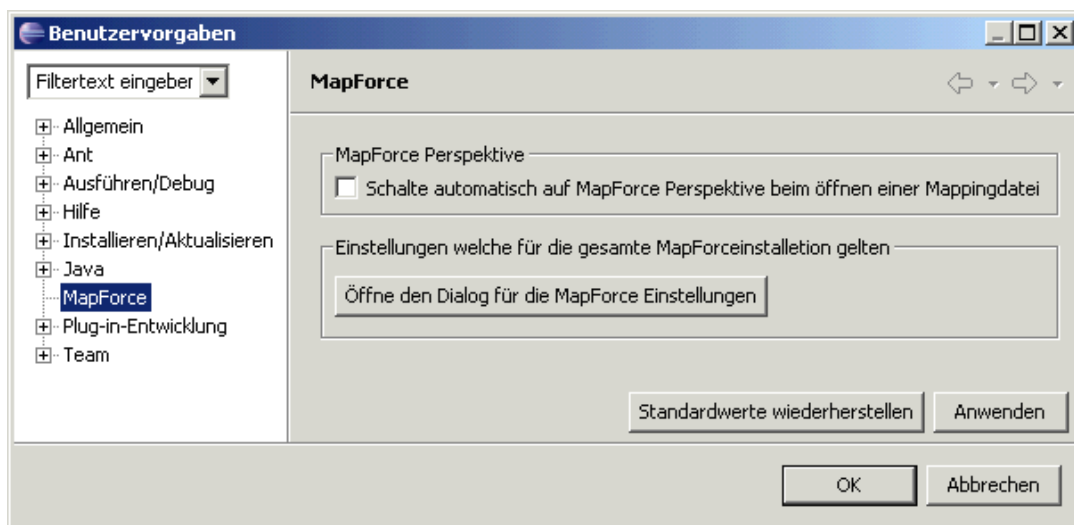
### Starten von Eclipse und Verwendung des MapForce Plug-in:

Nachdem Sie den MapForce for Eclipse Installer ausgeführt haben, sehen Sie eine leere Eclipse-Umgebung.



#### MapForce Eigenschaften:

1. Wählen Sie die Menüoption **Fenster | Benutzervorgabe** und klicken Sie auf den Eintrag MapForce.
2. Aktivieren Sie das Kontrollkästchen, das Sie sehen, um beim Öffnen einer Datei zur MapForce-Perspektive zu wechseln.

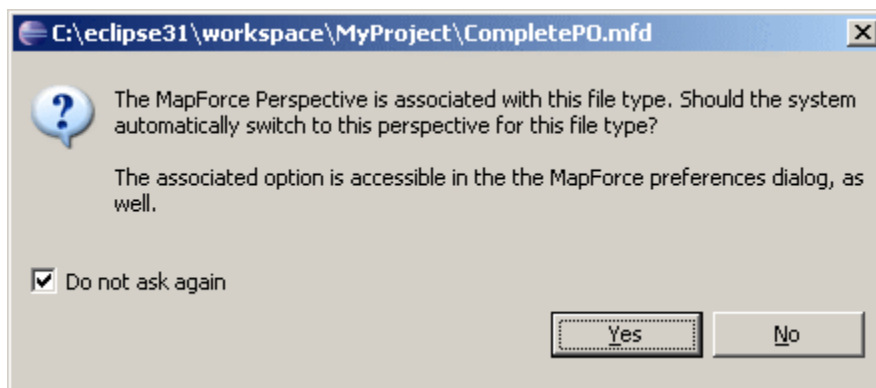


Wenn Sie auf die Schaltfläche "Öffne den Dialog für die MapForce Einstellungen" klicken, wird das Dialogfeld "Optionen" geöffnet, in dem Sie die jeweiligen MapForce Einstellungen, wie z.B. Bibliotheken, Codegenerierung usw. definieren können.

Bei Doppelklick auf eine MapForce Mapping-Datei (\*.mfd) wird zunächst ein Meldungsfeld angezeigt, in dem Sie darauf aufmerksam gemacht werden, dass eine MapForce Perspektive mit dieser Art von Datei verknüpft ist, und Sie werden gefragt, ob Eclipse in Zukunft automatisch in die MapForce Perspektive wechseln soll. Diese Einstellungen können später über die Option **Fenster | Benutzervorgaben | MapForce**



| MapForce Perspektive geändert werden.

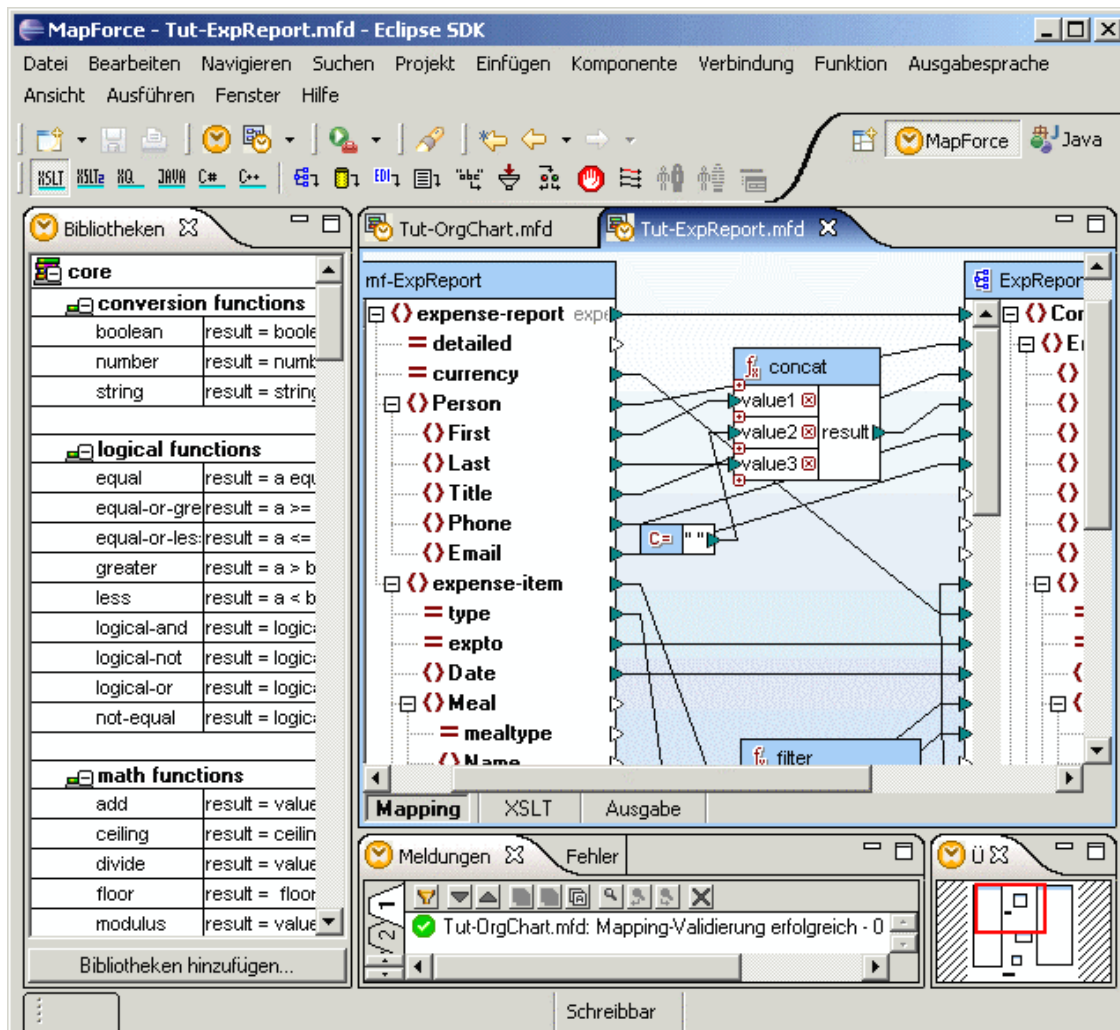


## 29.4 MapForce / Editor, Ansicht und Perspektiven

Sie können die MapForce Perspektive automatisch einstellen, wenn Sie im Dialogfeld **Window | Preferences** die Option "Automatically switch to MapForce perspective at file open" aktivieren. Um die Perspektive zu aktivieren, können Sie auch die unten beschriebene Option wählen.

**So aktivieren Sie die MapForce Perspektive in Eclipse:**

- Wählen Sie die Menüoption **Fenster | Perspektive öffnen | Andere | MapForce**. (In der Abbildung unten wird das Tut-ExpReport.mfd Mapping gezeigt.)



Die einzelnen MapForce-Register werden nun in der Eclipse-Umgebung angezeigt:

- das Register **Bibliotheken** auf der linken Seite, erlaubt Ihnen, vordefinierte oder benutzerdefinierte Funktionen auszuwählen.
- auf dem Register **Meldungen** werden Validierungsmeldungen, Fehler und Warnungen angezeigt
- auf dem Register **Übersicht** sehen Sie eine schematische Darstellung der Mapping-Datei.

Der Arbeitsbereich ist der Bereich, in dem Sie Ihre Mappings erstellen und eine Vorschau der Ausgabe ansehen können. Er besteht aus den folgenden Registern:

**Mapping** - hier sehen Sie das grafische Mapping-Design.

**XSLT** - hier sehen Sie den generierten XSLT-Code. Der Name dieses Registers entspricht der Option, die Sie unter Ausgabe | XSLT 1.0, XSLT 2.0 oder XQuery ausgewählt haben.

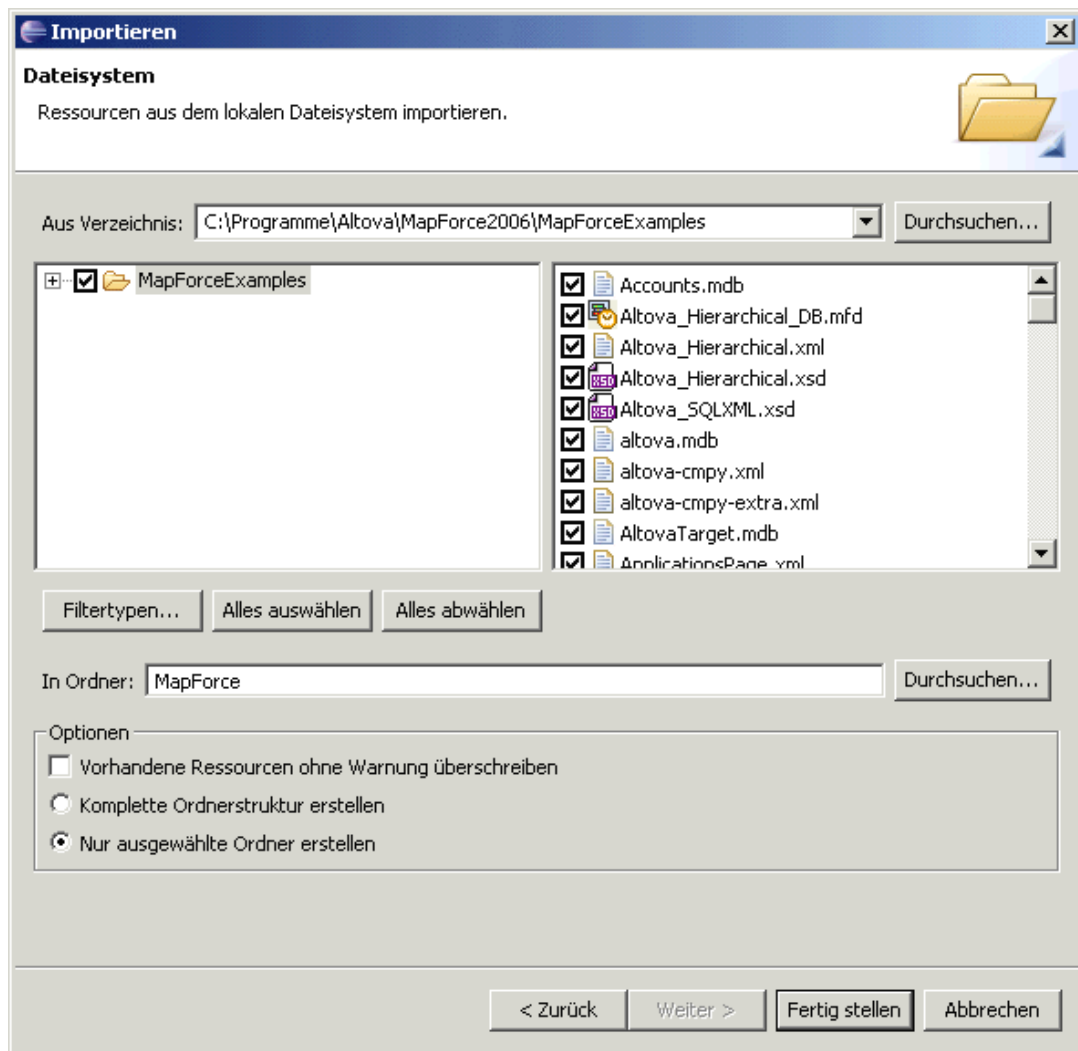
**Ausgabe-Vorschau** - hier sehen Sie die Mapping-Ausgabe, in diesem Fall die XML-Daten.

## 29.5 Importieren des Ordners MapForce examples in den Navigator

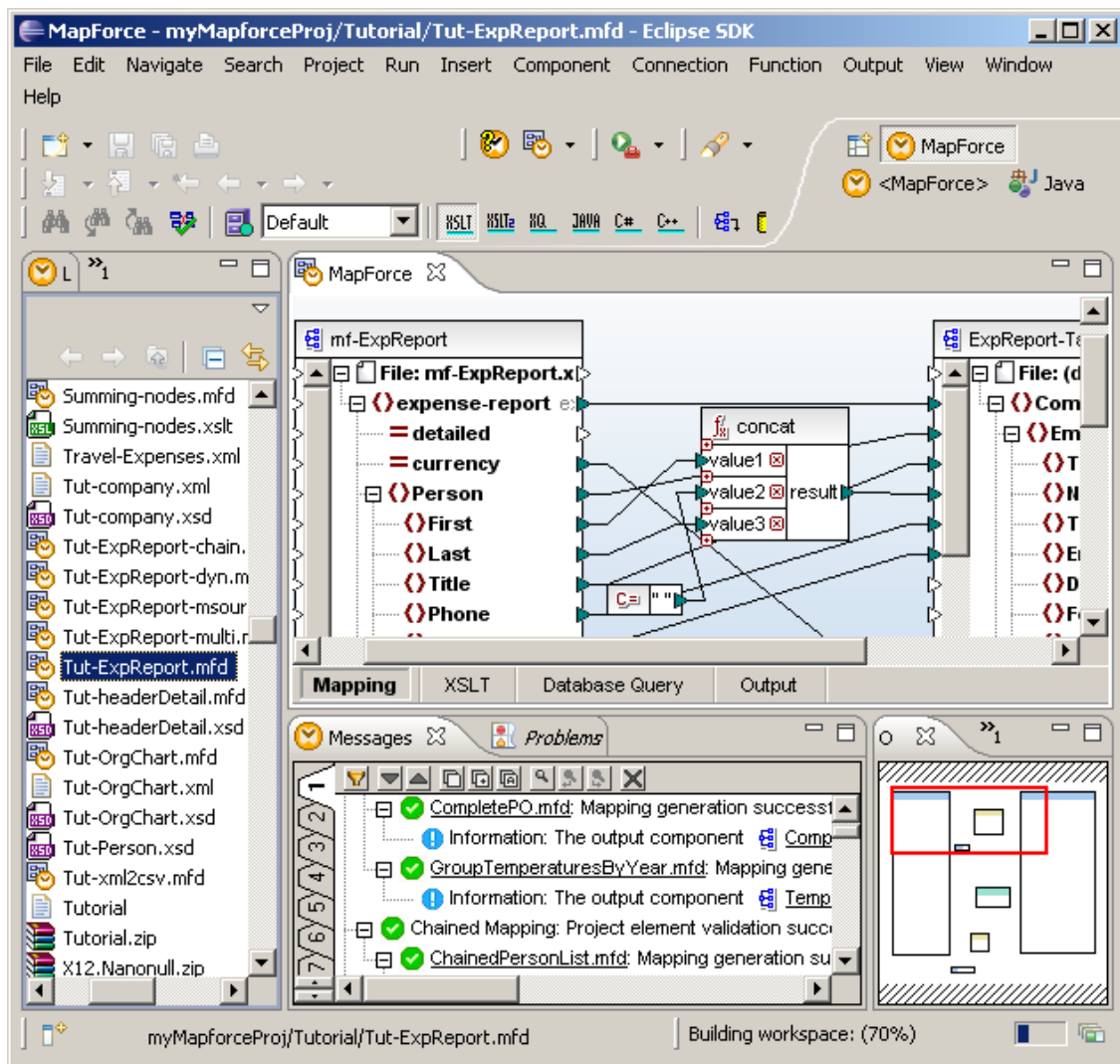
### So importieren Sie den Ordner MapForce Examples in den Navigator:

Wenn es noch kein Projekt gibt, erstellen Sie eines mit dem Befehl Datei | Neu | Projekt | MapForce/Eclipse.

1. Rechtsklicken Sie auf den Projektnamen auf dem Register **Navigator** und klicken Sie auf **Importieren**.
2. Wählen Sie den Eintrag "Allgemein | Dateisystem" und klicken Sie anschließend auf **Weiter**.
3. Klicken Sie auf die Schaltfläche **Durchsuchen** rechts vom Textfeld "Aus Verzeichnis:" und wählen Sie den Ordner MapForceExamples in Ihrem MapForce Verzeichnis aus.
4. Klicken Sie auf das Kontrollkästchen **MapForceExamples**.  
Dadurch wählen Sie alle Dateien in den verschiedenen Unterverzeichnissen im Fenster auf der rechten Seite aus.



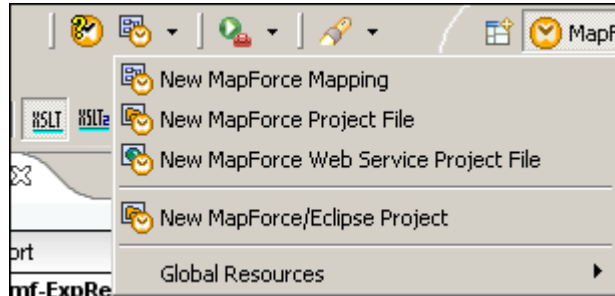
5. Falls der Zielordner nicht automatisch vorgeschlagen wird, klicken Sie auf die Schaltfläche **Durchsuchen** neben dem Textfeld "In Ordner:", um den Zielordner auszuwählen und klicken Sie auf **Fertig stellen**.  
Die ausgewählte Ordner- und Dateistruktur wird in den Eclipse-Arbeitsbereich kopiert.
6. Doppelklicken Sie auf eine Datei im Navigator, um sie zu öffnen.



## 29.6 Erstellen neuer MapForce Dateien (Mapping- und Projektdatei)

So erstellen Sie eine neue MapForce Mapping- oder Projektdatei:

- Klicken Sie auf die Auswahlliste "Neues MapForce Mapping" und wählen Sie die gewünschte Option aus.



- Neues MapForce Mapping, erstellt eine einzelne Mapping-Datei.
- Neue MapForce Projektdatei, erstellt ein MapForce Projekt, das mehrere Mappings zu einer Codegenerierungseinheit kombinieren kann.
- Neue MapForce Webservice Projektdatei, erstellt ein Webservice-Projekt.
- Neues MapForce Projekt, erstellt ein neues MapForce/Eclipse-Projekt und fügt den Ordner zum Navigatorfenster hinzu. Neue MapForce/Eclipse Projekte sind Eclipse-Projekte, denen ein MapForce Builder zugewiesen wurde. Nähere Informationen siehe [Verwendung von MapForce Eclipse-Projekten zur automatischen Erstellung eines Build](#)

## 29.7 MapForce Codegenerierung

### Build Integration

MapForce Mappings können in jedem Eclipse-Projekt enthalten sein. Die Generierung von Mapping-Code für das Mapping oder die MapForce-Projektdatei kann manuell durch Auswahl eines der Befehle aus dem Menü '**Code generieren...**' gestartet werden. Die komplette Integration in den automatischen Build-Erstellungsvorgang von Eclipse erreichen Sie durch Zuweisen des MapForce Builder zu einem Eclipse Projekt.

Informationen zur manuellen Codegenerierung finden Sie unter [Manuelles Erstellen von Mapping-Code](#)

Informationen zur automatischen Generierung von Mapping Code finden Sie unter [Verwenden von MapForce Eclipse zur automatischen Erstellung eines Build](#) und [Hinzufügen der MapForce Nature zum vorhandenen Eclipse-Projekt](#)

### 29.7.1 Manuelles Erstellen von Mapping-Code

#### So erstellen Sie Mapping-Code manuell für ein einzelnes Mapping:

1. Öffnen Sie das Mapping oder wählen Sie es aus und wählen Sie den Befehl **Datei | Code generieren in** oder **Datei | Code in ausgewählter Sprache generieren**. Sie werden aufgefordert, einen Zielordner für den generierten Code zu definieren.
2. Wählen Sie den Ordner aus und klicken Sie auf OK, um die Codegenerierung zu starten.  
Etwaige Fehler oder Warnmeldungen werden auf dem MapForce Meldungs-Register angezeigt.

#### So erstellen Sie manuell Mapping-Code für mehrere Mappings, die in einem einzigen MapForce Projekt miteinander kombiniert sind:

1. Öffnen Sie die MapForce Projektdatei bzw. wählen Sie sie aus.
2. Wählen Sie den Root Node oder einen beliebigen anderen Node im Projektdokument aus.
3. Wählen Sie im Kontextmenü den Befehl **Code generieren** oder **Code generieren in**. Der Zielordner für den generierten Code wird durch die Eigenschaften des ausgewählten Node bzw. die Eigenschaften des Parent-Node bestimmt.
4. Fehler oder Warnmeldungen werden auf dem MapForce Meldungs-Register angezeigt.



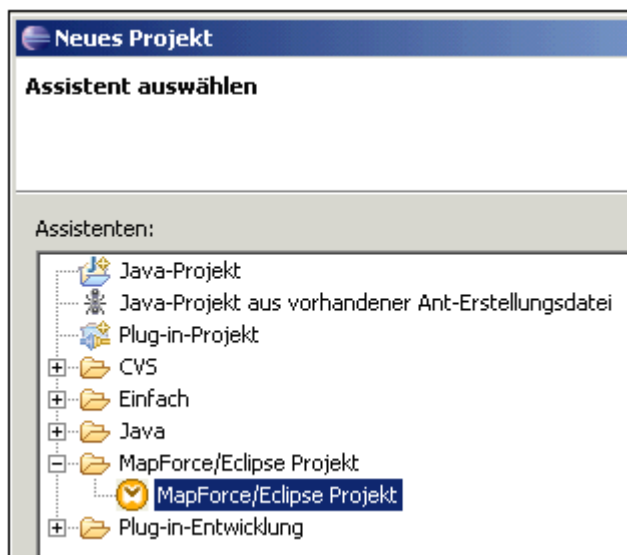
## 29.7.2

### Verwenden von MapForce Eclipse-Projekten zur automatischen Erstellung eines Build

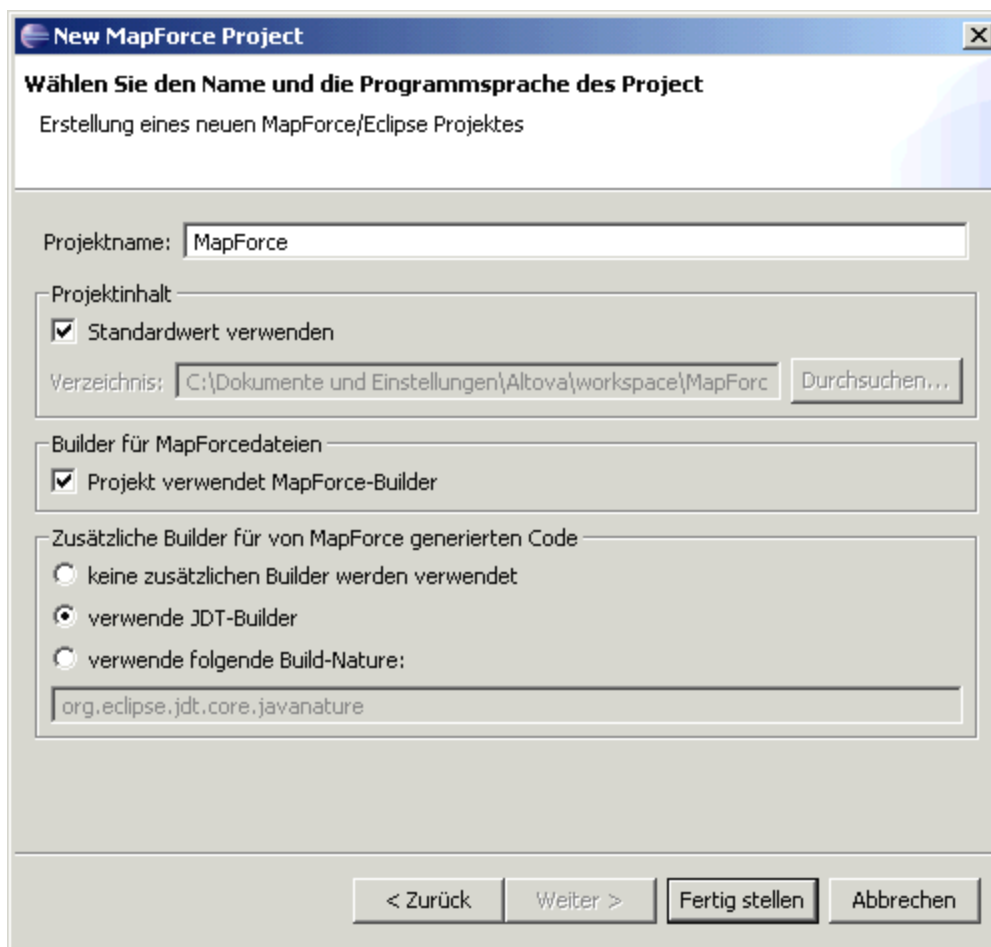
Das MapForce -Plug-in hat einen integrierten Projekt Builder. Dieser Builder wird durch die Projekt Nature-ID "com.altova.mapforceeclipseplugin.MapForceNature" identifiziert. MapForce Eclipse-Projekten wird diese Nature automatisch zugewiesen. Nähere Informationen dazu finden Sie unter "[Hinzufügen der MapForce Nature zum vorhandenen Eclipse-Projekt](#)".

#### So erstellen Sie ein neues MapForce Eclipse-Projekt:

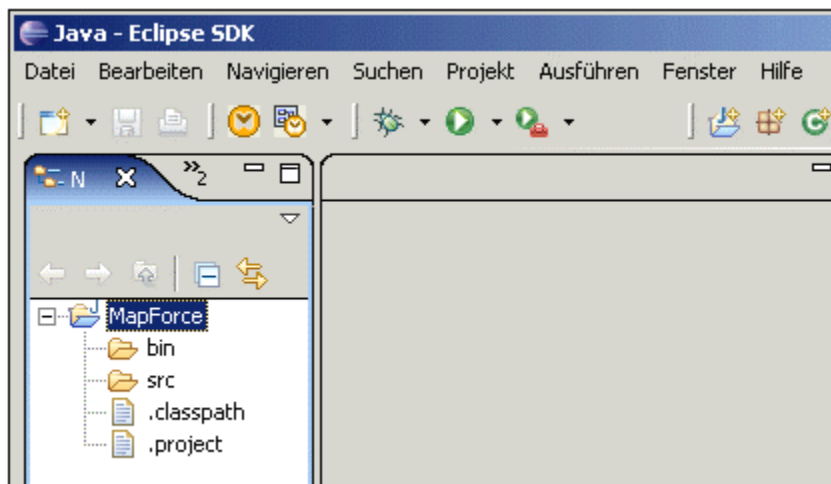
1. Klicken Sie auf das Register "Navigator", um es zum aktiven zu machen.
2. Rechtsklicken Sie in das Navigatorfenster und wählen Sie **Neu | Projekt**
3. Erweitern Sie den MapForce/Eclipse-Projekteintrag und wählen Sie MapForce/Eclipse. Klicken Sie anschließend auf "Weiter".



4. Geben Sie den Projektnamen ein (z.B. MapForce) und passen Sie gegebenenfalls die Projekteinstellungen an. Klicken Sie anschließend auf "Fertig stellen". Beachten Sie die Standardeinstellungen in der Gruppe "Zusätzliche Builder...". Verwenden Sie JTD builder.



Es wurde ein Eclipse-Projektordner und optional einige Ordner und Dateien innerhalb dieses Ordners erstellt.



Sie können nun innerhalb dieses Eclipse-Projekts MapForce Mappings und MapForce Projektdateien erstellen oder vorhandene hineinkopieren. Immer wenn sich ein Mapping oder eine MapForce Projektdatei ändert, wird der entsprechende Mapping-Code automatisch generiert. Fehler und Warnmeldungen, die bei der Codegenerierung erzeugt werden, werden in der MapForce Ansicht **Meldungen**

angezeigt und zur **Fehler**-Ansicht in Eclipse hinzugefügt.

Ein MapForce Eclipse-Projekt ist ein Eclipse Projekt, dem die MapForce Nature zugewiesen wurde, daher wird hierfür der MapForce Builder verwendet.

Wenn es im Eclipse-Projekt eine oder mehrere MapForce Projektdateien gibt, werden die Codegenerierungssprache und die Zielordner für die Ausgabe durch die Einstellungen in diesen Dateien festgelegt.

Auswirkungen, wenn eine MapForce Projektdatei im Eclipse-Projekt nicht vorhanden ist:

Wenn dem Projekt eine JDT-Nature zugewiesen wurde:

- In diesem Fall wird standardmäßig bei der Codegenerierung Java-Code erzeugt und das Java-Quellverzeichnis wird als Ausgabeverzeichnis für den Mapping-Code verwendet.

Beim Speichern eines Mappings wird automatisch Mapping-Code in Java generiert und kompiliert. Mit Hilfe des Befehls zum Debuggen oder Ausführen von Java-Code können Sie die erzeugte Mapping-Applikation testen.

Wenn dem Projekt **keine** JDT-Nature zugewiesen wurde:

- In diesem Fall ist das Zielverzeichnis für die Ausgabe der Projektordner und die Codegenerierungssprache wird entsprechend den aktuellen unter MapForce Optionen vorgenommenen Einstellungen gewählt.

#### So aktivieren Sie den automatischen Build-Vorgang:

1. Stellen Sie sicher, dass die Menüoption **Projekt | Automatisch erstellen** aktiviert ist.

#### So deaktivieren Sie die automatische Erzeugung von MapForce Mappingcode vorübergehend:

Diese Funktion steht nur bei Eclipse-Projekten zur Verfügung, zu denen die MapForce Nature hinzugefügt wurde.

1. Rechtsklicken Sie im Navigatorfenster auf die MapForce Projektdatei, \*.mfp.
2. Wählen Sie im Kontextmenü den Befehl **Eigenschaften**.
3. Klicken Sie im linken Bereich des Dialogfelds "Projekteigenschaften" auf den Eintrag "Erstellungsprogramme".
4. Deaktivieren Sie das Kontrollkästchen **MapForce Builder** im rechten Bereich.  
Bei Änderungen in Mapping-Dateien oder MapForce Projektdateien wird nun nicht mehr automatisch Mapping Code generiert.

### 29.7.3 Hinzufügen der MapForce Nature zum vorhandenen Eclipse-Projekt

**Anwenden der MapForce Nature auf bestehende Projekte:**

Fügen Sie den folgenden Text zum Abschnitt "Natures" der **.project**-Datei im MapForceOrdner hinzu, (z.B. im Ordner c:\eclipse33\workspace\MapForce\):

```
<nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>.
```

```
<natures>
 <nature>org.eclipse.jdt.core.javanature</nature>
 <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>
</natures>
```

Alle in diesem Projekt enthaltenen MapForce Projektdateien und Mappings werden nun beim automatischen Build berücksichtigt. Informationen zu MapForce-spezifischen Details finden Sie unter [Verwenden von MapForce Eclipse-Projekten zur automatischen Erstellung eines Build](#).

## 29.8 Erweitern des MapForce Plug-in

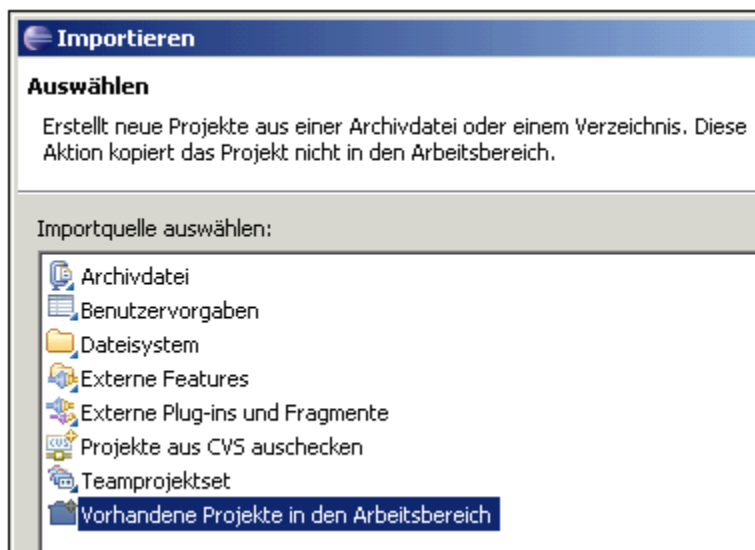
Das MapForce Plug-in bietet einen Erweiterungspunkt für Eclipse mit der ID "com.altova.mapforceeclipseplugin.MapForceAPI". Über diesen Erweiterungspunkt (Extension Point) können Sie die Funktionalitäten des MapForce Plug-in anpassen oder erweitern. Über den Erweiterungspunkt haben Sie Zugriff auf die COM-Schnittstelle des [MapForce Control](#) und der [MapForceAPI](#).

Ihr MapForce Eclipse Installationspaket enthält ein einfaches Beispiel für ein Plug-in, das diesen Erweiterungspunkt verwendet. Es überprüft, ob ein neues MapForce Mapping ein File open-Ereignis enthält und setzt den Zoomfaktor der Mapping-Ansicht auf 70%.

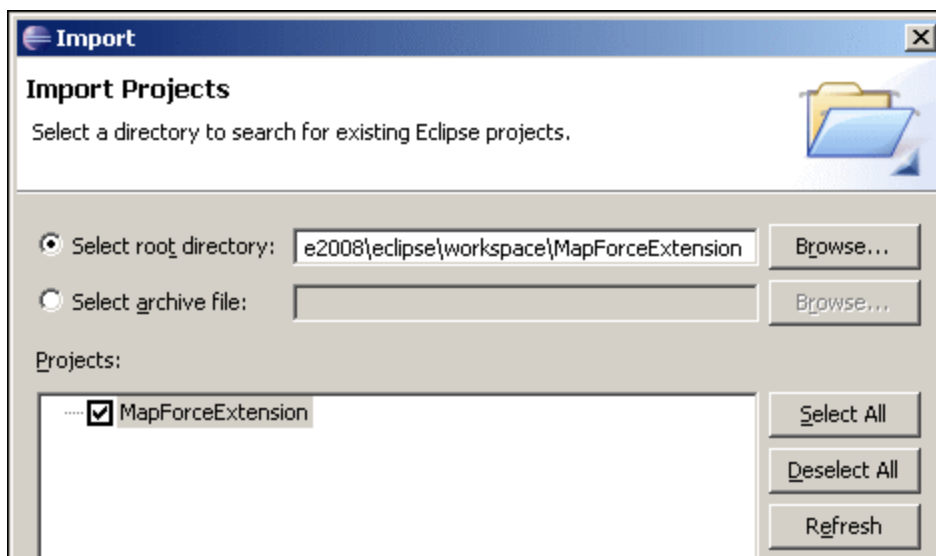
### Installieren des Beispiel-Erweiterungs-Plug-in:

Für das MapForce Plug-in muss das JDT (Java Development Tools) Plug-in installiert sein.

1. Starten Sie Eclipse.
2. Rechtsklicken Sie in den **Navigators** oder den PackageExplorer und wählen Sie den Menüeintrag **Importieren**.
3. Wählen Sie den Befehl "Vorhandene Projekte in den Arbeitsbereich" und klicken Sie auf "Weiter".



4. Klicken Sie auf die Schaltfläche **Durchsuchen...** neben dem Feld "'Stammverzeichnis auswählen'" und wählen Sie das Verzeichnis des Beispielprojekts z.B. **C:\Program Files\Altova\MapForce2012\eclipse\workspace\MapForceExtension**).



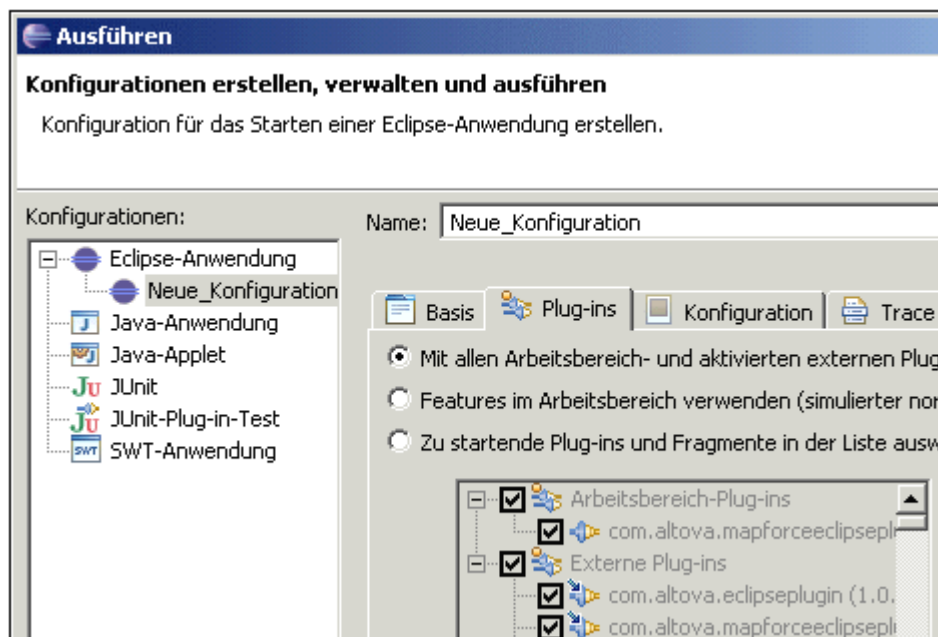
5. Klicken Sie auf "Fertig stellen".  
In Ihrem Arbeitsbereich wurde nun ein neues Projekt namens "MapForceExtension" erstellt.

#### Aufrufen von javadoc- für den Erweiterungspunkt des MapForce Plug-in:

1. Öffnen Sie die Datei **index.html** im Ordner docs der Plug-Installation z.B c:\Program Files\Altova\MapForce2012\eclipse\plugins\com.altova.mapforceeclipseplugin\_11.0.0\docs\

#### Ausführen des Beispiel-Erweiterungs-Plug-in:

1. Wechseln Sie zur Java-Perspektive.
2. Wählen Sie die Menüoption **Ausführen | Ausführen...**
3. Wählen Sie den Eintrag Eclipse Anwendung und klicken Sie auf **Neue\_Konfiguration**.



4. Stellen Sie sicher, dass der MapForceClient für das Projekt auf dem Register 'Plug-ins' ausgewählt ist.

5. Klicken Sie auf die Schaltfläche "Ausführen".  
Eine neue Eclipse Workbench wird geöffnet.
6. Öffnen Sie ein beliebiges MapForce Mapping in der neuen Workbench. Es wird mit einem Zoomfaktor von 70 % geöffnet.





# Kapitel 30

---

Benutzerreferenz

## 30 Benutzerreferenz

Der folgende Abschnitt enthält eine Liste aller Menüs und Menüoptionen von MapForce sowie eine kurze Beschreibung davon.

## 30.1 Datei

### Neu

Erstellen ein neues Mapping-Dokument oder Mapping-Projekt

### Öffnen

Öffnet zuvor gespeicherte Mapping (\*.mfd) oder Mapping-Projekt (mfp)-Dateien.

Bitte beachten Sie:

Ein Mapping, das Funktionen enthält, die nur in einer höheren MapForce Edition verfügbar sind, kann nicht geöffnet werden.

So kann z.B. ein Mapping, das Webservice-Funktionen enthält in der Professional Edition nicht geöffnet werden. Datenbankmappings sind z.B. in Basic Editions nicht möglich.

### Speichern

Speichert das aktuelle Mapping unter dem gerade aktiven Dateinamen.

### Speichern unter

Speichert das aktuelle Mapping unter einem anderen Namen oder ermöglicht Ihnen, dem Projekt einen neuen Namen zu geben, falls Sie es zum ersten Mal speichern.

### Alles speichern

Speichert alle derzeit geöffneten Mapping-Dateien.

### Neu laden

Lädt die aktuell aktive Mapping-Datei neu. Sie werden gefragt, ob Ihre letzten Änderungen verworfen werden sollen.

### Schließen

Schließt die aktuell aktive Mapping-Datei. Sie werden gefragt, ob Sie die Datei speichern möchten, bevor sie sie schließen.

### Alles schließen

Schließt alle derzeit geöffneten Mapping-Dateien. Sie werden gefragt, ob Sie eine der nicht gespeicherten Mapping-Dateien speichern möchten.

### Projekt speichern

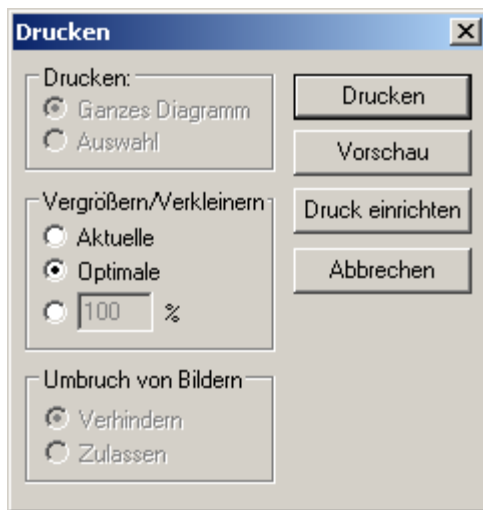
Speichert das aktuell aktive Projekt.

### Projekt schließen

Schließt das aktuell aktive Projekt.

### Drucken

Öffnet das Dialogfeld Drucken, über das Sie Ihr Mapping ausdrucken können.



Mit "Aktuelle" wird der aktuell definierte Zoom-Faktor des Mappings beibehalten. Bei Auswahl von "Optimale" wird das Mapping auf Seitengröße vergrößert/verkleinert. Sie können auch einen numerischen Zoom-Faktor angeben. Die Bildlaufleisten der Komponente werden nicht gedruckt. Außerdem können Sie festlegen, ob die Grafik auf mehrere Seiten umbrochen werden soll oder nicht.

**Druckvorschau**

Öffnet dasselbe Dialogfeld "Drucken" mit denselben Einstellungen wie oben beschrieben.

**Druckereinrichtung**

Öffnet das Dialogfeld "Druckereinrichtung", in dem Sie den gewünschten Drucker und die Seitenausrichtung usw. einstellen können.

**Mapping validieren**

Beim Validieren eines Mappings wird überprüft, ob alle Mappings (Konnektoren) gültig sind und entsprechende Warn- oder Fehlermeldungen werden ggf. angezeigt. Nähere Informationen dazu finden Sie unter "[Validieren von Mappings](#)".

**Mapping-Einstellungen**

**Mapping-Einstellungen**

Mapping-Output  
Applikationsname: Mapping

Java-Einstellungen  
Base Package Name: com.mapforce

Dateipfadeinstellungen  
☒ Pfade im generierten Code absolut machen  
☒ Windows-Pfadkonvention für Dateipfadausgabe für Dateien von einem lokalen Dateisystem verwenden

WebService Operation - Einstellungen  
WSDL-Definitionen: query.wsdl  
Service: {http://www.altova.com/WS2DB/query.wsdl}-WS2DB  
Endpoint: WS2DBSoapPort  
Operation: {http://www.altova.com/WS2DB/query.wsdl}-getPerson

OK Abbrechen

Hier sind die dokumentspezifischen Einstellungen definiert. Sie werden in der \*.mfd Datei gespeichert.

#### *Mapping Output*

Applikationsname: definiert das Präfix für den XSLT1.0/2.0-Dateinamen oder den Java, C# oder C++ Applikationsnamen für die generierten Transformationsdateien.

#### *Java-Einstellungen*

Base Package Name: definiert den Base Package-Namen für die Java-Ausgabe.

#### *Dateipfadeinstellungen*

##### **Pfade im generierten Code absolut machen**

Damit wird sichergestellt, dass der generierte Code mit Mapping-Dateien (\*.mfd) aus Versionen vor Version 210 kompatibel ist. Nähere Informationen dazu finden Sie unter [Relative und absolute Dateipfade](#).

##### **Windows-Pfadkonvention für Dateipfadausgabe ... verwenden**

Mit Hilfe dieses Kontrollkästchens stellen Sie sicher, dass die Windows-Pfadkonventionen erfüllt werden. Bei der Ausgabe von XSLT2 (und XQuery) wird der gerade verarbeitete Dateiname intern mit der Funktion document-uri abgerufen. Diese Funktion gibt einen Pfad in der Form Datei:// URI für lokale Dateien zurück.

Wenn dieses Kontrollkästchen aktiviert ist, wird eine Datei:// URI Pfadspezifikation zur einfacheren Weiterverarbeitung automatisch in einen vollständigen Windows-Dateipfad (z.B. "C:\...") konvertiert.

#### *WebService Operation - Einstellungen*

Die Felder "WSDL-Definitionen", "Service", "Port" und "Operation" werden automatisch ausgefüllt, wenn das Mapping-Dokument Teil einer Webservice-Implementierung ist.

### **Code in ausgewählter Sprache generieren**

Generiert Code in der aktuell ausgewählten Sprache Ihres Mappings. Die aktuell ausgewählte Sprache wird durch ein markiertes Symbol in der Titelleiste angezeigt: XSLT, XSLT2, XQuery, Java, C# oder C++.

### **Code generieren in | XSLT/XSLT2**

Mit diesem Befehl wird/werden die für die Transformation aus der/den Quelldatei(en) benötigte(n) XSLT-Datei(en) generiert.  
Nach Auswahl dieser Option wird das Dialogfeld zum Suchen des Ordners geöffnet, in dem Sie den Speicherort der XSLT-Datei angeben können.

Hinweis: Der Name der generierten XSLT-Datei(en) ist im Feld **Anwendungsname** des Dialogfelds "Mapping-Output" definiert. Dieses Dialogfeld lässt sich durch Auswahl von **Datei | Mapping-Einstellungen** öffnen.

### **Code generieren in | XQuery**

Mit diesem Befehl wird/werden die für die Transformation aus der/den Quelldatei(en) benötigte(n) XQuery-Datei(en) generiert.  
Nach Auswahl dieser Option wird das Dialogfeld zum Suchen des Ordners geöffnet, in dem Sie den Speicherort der XQuery-Datei angeben können.

Hinweis: Der Name der generierten XQuery-Datei(en) wird im Dialogfeld "Mapping-Output" im Feld **Anwendungsname** definiert. Um dieses Dialogfeld zu öffnen, wählen Sie die Menüoption **Datei | Mapping-Einstellungen**.

### **Code generieren in | Java**

### **Code generieren in | C#**

### **Code generieren in | C++**

Mit diesen Befehlen wird/werden die für ein vollständiges Applikationsprogramm benötigte(n) Java-Datei(en) generiert.  
Nach Auswahl dieser Option wird das Dialogfeld zum Suchen des Ordners geöffnet, in dem Sie den Pfad der generierten Dateien auswählen können.

Hinweis: Die Namen der generierten Applikationsdateien (sowie die Projektdaten: \*.csproj C#-Projektdatei, \*.sln Lösungsdatei, \*.vcproj Visual C++-Projektdatei) sind im Feld **Anwendungsname** des Dialogfelds "Mapping Output" definiert. Dieses Dialogfeld lässt sich durch Auswahl von **Datei | Mapping-Einstellungen** öffnen.

Der durch den ausgeführten Code erstellte Dateiname ist der Name, der im Feld "XML-Output-Instanz" (für die Codegenerierung) des Dialogfelds [Komponenteneinstellungen](#) angezeigt wird, wenn es sich beim Ziel um ein XML/Schema-Dokument handelt.

### **Dokument generieren**

Generiert detaillierte Dokumentation zu Ihren Mapping-Projekten in verschiedenen Ausgabeformaten. Nähere Informationen dazu finden Sie unter [Dokumentieren von Mapping-Projekten](#).

### **Letzte Dateien**

Zeigt eine Liste der zuletzt geöffneten Dateien an.

### **Letzte Projekte**

Zeigt eine Liste der zuletzt geöffneten Projekte an.

**Beenden**

Beendet die Applikation. Sie werden gefragt, ob Sie nicht gespeicherte Dateien speichern möchten.

## 30.2 Bearbeiten

Die meisten der Befehle in diesem Menü werden erst aktiv, wenn Sie das Ergebnis eines Mappings im **Ausgabe**-Fenster bzw. z.B. den XSLT-Code im XSLT-Fenster ansehen.

### Rückgängig



In MapForce steht eine unbegrenzte Anzahl an "Rückgängig"-Schritten zur Verfügung, mit denen Sie Ihr Mapping Schritt für Schritt wieder rückgängig machen können.

### Wiederherstellen



Mit dem Befehl "Wiederherstellen" können Sie zuvor rückgängig gemachte Befehle wiederherstellen. Auf diese Weise können Sie mit Hilfe dieser beiden Befehle den Verlauf Ihres Mappings Schritt für Schritt vor und zurück durchgehen.

### Suchen



Dient zum Suchen von bestimmtem Text auf der Registerkarte XSLT, XLST2, XQuery oder Ausgabe.

### Weitersuchen F3



Sucht nach der nächsten Stelle, an der der Such-String vorkommt.

### Ausschneiden/Kopieren/Einfügen/Löschen

Mit den Windows-Standardbearbeitungsbefehlen können Sie beliebige im Mapping-Fenster angezeigte Komponenten oder Funktionen ausschneiden, kopieren usw.

### Alles markieren

Markiert alle Komponenten auf dem Register "Mapping" oder den Text/Code auf den Registern XSLT, XSLT2, XQuery bzw. Ausgabe.



## 30.3 Einfügen

### XML Schema / Datei

Fügt eine XML-Schemadatei als Datenquelle oder Zielkomponente in das Mapping-Fenster ein. Sie können XML-Dateien mit einer Schemareferenz auswählen. In diesem Fall wird das referenzierte Schema automatisch eingefügt. Wenn Sie eine XML-Schema-Datei einfügen, werden Sie gefragt, ob Sie eine XML-Instanzdatei bereitstellen möchten, die die Daten für die XSLT-, XSLT2, XQuery und Ausgabe enthält. Wenn Sie eine XML-Datei ohne eine Schemareferenz auswählen, werden Sie gefragt, ob automatisch ein XML-Schema dafür generiert werden soll.

### Datenbank

Fügt eine Schemakomponente als Datenquelle oder Zielkomponente ein. Die Datenbank liefert die Daten für die Schemastruktur und zeigt sie in einer Baumstrukturansicht an.

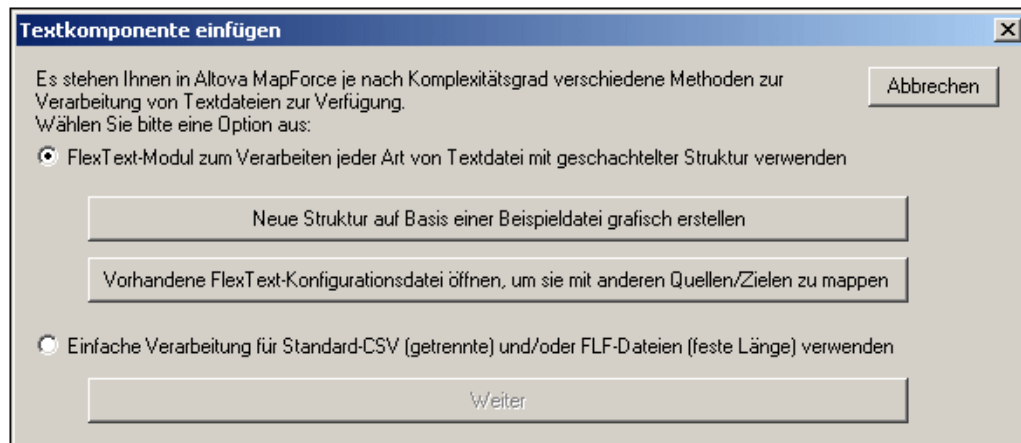
### EDI

Fügt ein EDI-Dokument ein, das als Datenquelle oder als Zielkomponente verwendet werden kann. Nähere Informationen dazu finden Sie im Abschnitt "[EDI](#)".

### Textdatei

Daraufhin wird das Dialogfeld "Textkomponente einfügen" geöffnet, in dem Sie auswählen können, ob Sie:

- eine Beispieltextdatei für die Erstellung der neuen FlexText-Komponente auswählen möchten,
- eine vorhandene FlexText-Datei öffnen möchten, um sie auf andere Zielkomponenten zu mappen
- oder eine Standard-CSV- oder FLF-Textdatei einfügen möchten.



### Webservice-Funktion

Fügt eine Webservice-Funktion in ein Mapping ein, die durch eine WSDL-Datei definiert ist. Wenn Sie Input- und Output-Komponenten damit verbinden, können Sie im Ausgabefenster sofort eine Vorschau des Ergebnisses der Webservice-Funktion ansehen. Nähere Informationen dazu siehe "[Aufrufen von Webservices](#)".

### Excel 2007-Datei

Fügt Microsoft Excel 2007 (und höher) (Office Open XML)-Dateien (\*.xlsx) sowohl als Quell- als auch als Zielkomponente ein. Nähere Informationen dazu siehe: [Mappen von MS OOXML Excel 2007 \(und höher\)-Dateien](#).

Microsoft Excel 2007 (und höher) muss nur installiert werden, wenn Sie Excel 2007 (und höher) Blätter auf dem Register "Ausgabe" ansehen möchten. Wenn Sie eine frühere Version von Microsoft Excel verwenden oder Excel nicht installiert haben, können Sie Excel 2007 (und höher)-Dateien trotzdem mappen. Sie können das Ergebnis des Mappings aber nicht auf dem Ausgaberegister anzeigen. Sie können es jedoch speichern, indem Sie auf die Schaltfläche "Ausgabe speichern" klicken.

### XBRL-Dokument

Fügt eine XBRL-Instanz oder ein Taxonomiedokument ein. Nähere Informationen zum Mappen von Taxonomien und Instanzdateien finden Sie unter [Mappen von XBRL](#).

### Konstante

Fügt eine Konstante als Funktionskomponente ein, die Fixdaten für ein Input-Symbol bereitstellt. Die Daten werden in das Dialogfeld eingegeben, wenn die Komponente erzeugt wird. Eine Konstantenfunktion hat nur ein Output-Symbol. Sie können die folgenden Datentypen wählen: Zahl und String.

### Variable

Fügt eine Zwischenvariable ein, die einer regulären (nicht-inline gesetzten) benutzerdefinierten Funktion entspricht. Variablen sind Strukturkomponenten ohne Instanzdateien und dienen zum Vereinfachen eines Mappings. Nähere Informationen dazu finden Sie unter [Zwischenvariablen](#).

### Filter: Nodes/Zeilen

Fügt eine Komponente ein, die zwei Eingabe- und zwei Ausgabeparameter verwendet: **node/row** und **bool** sowie **on-true** und **on-false**. Ist der Boolesche Wert "true", wird der Wert des node/row-Parameters an den on-true-Parameter übergeben. Wenn der Boolesche Wert "false" ist, wird der Komplementwert an den on-false-Parameter übergeben. Nähere Informationen zur Verwendung eines Filters finden Sie in dem im Tutorial beschriebenen Beispiel zur [Verwendung von Filtern](#).

### SQL-WHERE-Bedingung

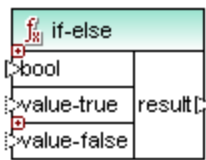
Fügt eine spezielle Filterkomponente für Datenbankdaten ein, mit der Sie jede beliebige SQL WHERE-Klausel an die von MapForce generierten Abfragen anhängen können. Nähere Informationen dazu finden Sie unter: [SQL WHERE-Komponente/Bedingung](#).

### Wertezuordnung

Fügt eine Komponente ein, die einen Input-Wert mittels einer Lookup-Tabelle in einen Output-Wert transformiert. Die Komponente hat ein Input- und ein Output-Datenelement. Nähere Informationen dazu finden Sie unter [Wertezuordnung - Transformieren von Input-Daten](#).

### IF-Else-Bedingung

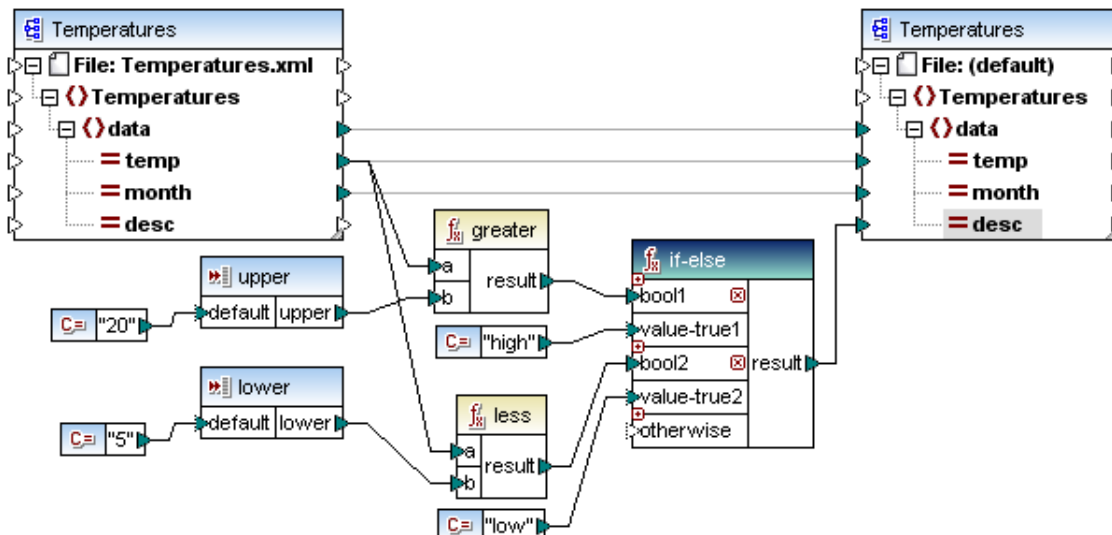
Eine Bedingung ist eine Komponente, die es ermöglicht, je nach Ergebnis der aktuellen Bedingung unterschiedliche Daten-Sets zu übergeben. In der Komponentenüberschrift wird der Text "if-else" angezeigt.



- Der erste Eingabeparameter "**bool**" ist ein **Boolescher Wert**. Er enthält die Daten, auf die das Datenelement überprüft wird.
- Der Eingabeparameter **value-true** enthält die Daten, die als Ergebnis übergeben werden sollen, falls die Bedingung "true" ist.
- **value-false** liefert die Daten, die übergeben werden, wenn die Bedingung "false" ist.
- Der **result**-Parameter gibt die Daten aus den Eingabeparametern value-true/false aus.

Die IF-Else Funktion ist **erweiterbar**. Dies bedeutet, Sie können die Daten auf mehrere Bedingungen überprüfen und die Else-Bedingung/den Else-Wert über den **otherwise** Parameter ausgeben.

Wenn Sie auf das Plus-Symbol klicken, wird ein neues if-else-Paar eingefügt, also boolX und value-trueX, während das Parameterpaar bei Klick auf das "x" gelöscht wird.



Im Beispiel oben werden die Temperaturdaten analysiert:

- Wenn temp **größer** als 20 ist, so wird der Wert "true" an **bool1** übergeben und das Ergebnis ist "**high**" von **value-true1**.
- Wenn temp **kleiner** als 5 ist, so wird der Wert "true" an **bool2** übergeben und das Ergebnis ist "**low**" von **value-true2**.
- Andernfalls ist nichts (eine leere Sequenz) das Ergebnis der Komponente, da es keine Verbindung zum "otherwise" Input gibt.

Ergebnis des Mappings:

```
<?xml version="1.0" encoding="UTF-8"?>
<Temperatures xsi:noNamespaceSchemaLocation="c:/DOCUME~1
http://www.w3.org/2001/XMLSchema-instance">
 <data temp="-3.6" month="2006-01" desc="low"/>
 <data temp="-0.7" month="2006-02" desc="low"/>
 <data temp="7.5" month="2006-03"/>
 <data temp="12.4" month="2006-04"/>
</Temperatures>
```

### Ausnahme



Mit Hilfe der Ausnahme-Komponente können Sie ein Mapping unterbrechen, wenn eine bestimmte Bedingung erfüllt wird, oder Sie können beim Arbeiten mit WSDL-Mapping-Projekten Fehlermeldungen definieren. Nähere Informationen finden Sie unter [MapForce Ausnahmeereignisse](#) oder [Webservice Fehler](#).

## 30.4 Projekt

MapForce unterstützt die Mehrfachdokumentenschnittstelle (MDI) und gestattet Ihnen, Ihre Mappings in Mapping-Projekte zu gruppieren. Projektdateien haben die Erweiterung **\*.mfp**.

Es können zwei Arten von Projekten definiert werden:

- eine Sammlung einzelner Mappings, also ein Standardprojekt.
  - eine Gruppe miteinander in Zusammenhang stehender Mappings, aus denen ein WSDL-Mapping-Projekt besteht, mit dem ein Webservice implementiert wird.
- Beide Arten unterstützen die Codegenerierung für das gesamte Projekt.

### Dateien zu Projekt hinzufügen:

Damit können Sie über das Dialogfeld "Öffnen" Mappings zum aktuellen Projekt hinzufügen.

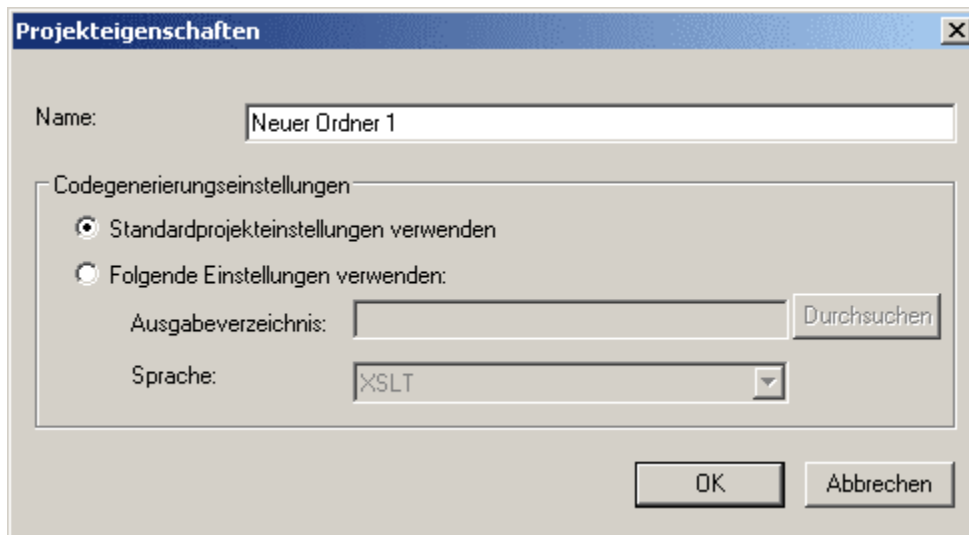
### Aktive Datei zu Projekt hinzufügen:

Fügt die aktuell aktive Datei zum aktuell geöffneten Projekt hinzu.

### Ordner erstellen:

Mit dieser Option wird ein neuer Ordner zur aktuellen Projektstruktur hinzugefügt. Diese Option wird nur dann aktiv, wenn dies möglich ist.

Sie können entweder die Standardprojekteinstellungen verwenden oder Ihre eigenen definieren, indem Sie auf die Schaltfläche "Folgende Einstellungen verwenden" klicken.



### Mapping öffnen:

Öffnet das gerade auf dem Register "Projekt" ausgewählte Mapping.

### Mapping für Operation erstellen:

Erstellt eine Mapping-Datei für die aktuell ausgewählte Operation des WSDL-Projekts. Der Name der Operation, der in der WSDL-Datei definiert ist, wird über das Dialogfeld "Speichern unter" bereitgestellt, das automatisch geöffnet wird.

### Mapping-Datei für Operation hinzufügen:

Mit dieser Option können Sie eine zuvor gespeicherte Mapping-Datei zur aktuell aktiven WSDL-Operation hinzufügen. Wählen Sie die Mapping-Datei im Dialogfeld "Öffnen" aus.

### Webservice einfügen...

Dient zum Einfügen eines auf einer bestehenden WSDL-Datei basierenden Webservices.

**Datei in XMLSpy öffnen**

Öffnet die ausgewählte, im Projektfenster markierte WSDL-Datei in XMLSpy.

**Code für das gesamte Projekt generieren:**

Generiert Projektcode für das gesamte im Projektfenster angezeigte Projekt. Der Code wird für alle Mapping-Dateien (\*.mfd) in den einzelnen Ordnern in der aktuell ausgewählten Standardsprache generiert.

**Code generieren in...**

Generiert Projektcode in der Sprache, die Sie aus dem Submenü auswählen.

**Projekt neu laden**

Lädt das aktuell aktive Projekt neu und wechselt zum Register "Projekt".

**Projekteigenschaften:**

Klicken Sie im Projektfenster auf den Projektnamen (z. B. MapForceExamples) und wählen Sie den Befehl Projekt | Eigenschaften. Im Dialogfeld können Sie die projektweiten Eigenschaften definieren. Wenn Sie im Projektfenster auf einen Ordner oder einen Dateinamen klicken und diesen Befehl auswählen, wird ein Dialogfeld für das jeweilige Datenelement geöffnet.

**Projekteigenschaften**

Projektname:

Projekt-Verzeichnis:

Ausgabeeinstellungen

Ausgabename:

Ausgabeverzeichnis:

Sprache:  ▼

Java-Einstellungen

Base Package Name:

## 30.5 Komponente

### Root-Element ändern

Dient zum Ändern des Root-Elements des XML-Instanzdokuments.

### Schema-Definition in XMLSpy bearbeiten

Wenn Sie nach Auswahl eines XML-Schemas/ Dokuments auf diese Option klicken, wird eine XML-Schemadatei in der Schema-Ansicht von XMLSpy geöffnet, wo Sie sie bearbeiten können.

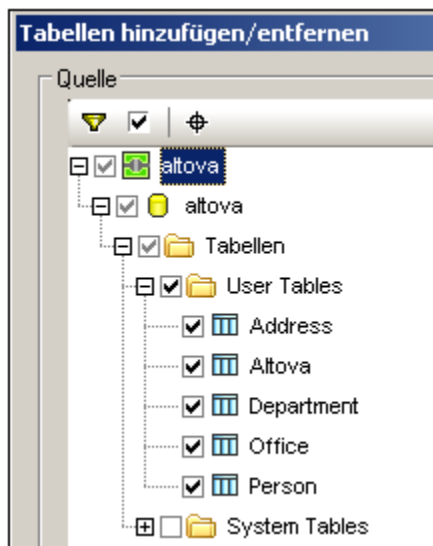
### FlexText-Konfiguration bearbeiten

Öffnet FlexText und gestattet Ihnen, eine zuvor erstellte FlexText-Datei zu bearbeiten.

### Tabellen hinzufügen/entfernen

Dient zum Ändern der Anzahl der Tabellen in der Datenbankkomponente.

Aktivieren/Deaktivieren Sie dazu den jeweiligen Eintrag in der Gruppe "Datenbanktabellen".



### Mapping auf EDI X12 997 erstellen

Das X12 997 Functional Acknowledgment gibt über den Status des EDI-Datenaustauschs Auskunft. Alle Fehler, die bei der Verarbeitung des Dokuments auftreten, werden hier ausgegeben.

MapForce kann im Hauptmappingbereich automatisch ein X12 997-Dokument generieren, das Sie an den Empfänger senden können. Nähere Informationen zu bestimmten Fehlersegmenten und deren Bedeutung finden Sie im [X12 997 Acknowledgment](#).

### Mapping auf EDI X12 999 erstellen

Das X12 999 Implementation Acknowledgment Transaction Set meldet die Nichterfüllung von HIPAA-Implementierungsrichtlinien oder Applikationsfehler. Als Antwort auf jede EDI Transaction, die an ein Unternehmen gesendet wird, muss eine 999 Transaction zurückgesendet werden. Nähere Informationen dazu finden Sie unter [X12 999 - Nichterfüllung der Implementierungsrichtlinien](#).

### Aktualisieren

Aktualisiert die gerade aktive Datenbankkomponente anhand der Datenbank.

### Duplikat erzeugen

Fügt eine Kopie/einen Klon des ausgewählten Elements ein und ermöglicht Ihnen dadurch,

mehrere Eingabedaten auf dieses Element zu mappen. Duplizierte Datenelemente haben kein Output-Symbol und können nicht als Quellelemente verwendet werden. Ein Beispiel dazu finden Sie im Tutorial im Abschnitt "[Duplizieren von Datenelementen](#)".

**Duplikat löschen**

Löscht ein zuvor definiertes Duplikat. Nähere Informationen dazu finden Sie im Tutorial im Abschnitt "[Duplizieren von Datenelementen](#)".

**Datenbankaktionen**

Dient zum Definieren der Tabellenaktionen, die mit den gemappten Daten an der jeweiligen Zieldatenbanktabelle durchgeführt werden sollen. Nähere Informationen dazu finden Sie im Abschnitt "[Tabellenaktionen, Schlüsseleinstellungen, Transaktionsverarbeitung](#)".

**Datenbank abfragen**

Erstellt anhand der Tabelle bzw. des Felds, die bzw. das Sie in der Datenbankkomponente ausgewählt haben, eine Select-Anweisung. Wenn Sie einmal auf eine Tabelle/ein Feld klicken, wird dieser Befehl aktiv und die Select-Anweisung wird automatisch in das Select-Fenster platziert.

**Links ausrichten**

Richtet die Datenelemente entlang des linken Fensterrands aus.

**Rechts ausrichten**

Richtet die Datenelemente entlang des rechten Fensterrands aus. Diese Anzeige eignet sich vor allem, um Mappings auf das Zielschema zu erzeugen.

**Eigenschaften**

Öffnet ein Dialogfeld, in dem die Einstellungen der aktuell ausgewählten Komponente angezeigt werden. Wenn es sich bei der Komponente um eine XML-Schema-Datei handelt, wird das Dialogfeld "Komponenteneinstellungen" geöffnet. Bei einer Textdatei wird das Dialogfeld "Text importieren / exportieren" geöffnet.

**Komponentenname:**

Alle dateibasierten (d.h. nicht auf einer Datenbank basierenden) Komponenten eines Mappings haben einen Komponentennamen, der bei der Erstellung der Komponente automatisch ausgefüllt wird. Sie können den Namen jedoch jederzeit ändern.

Über den Komponentennamen können spezifische Komponenten über die Befehlszeile (über den [/BUILTIN](#) Switch) oder über FlowForce aufgerufen werden. Damit die Komponente über die Befehlszeile oder über FlowForce aufgerufen werden kann, muss der Komponentename eindeutig sein.

Je nachdem, welche Art von Komponente eingefügt wird, wird der Standardname auf unterschiedliche Art generiert. Der Name kann auf folgenden Kriterien basieren:

- Name der XML-Input/Output-Datei
- Taxonomiename
- Name der EDI-Nachricht
- Name der FlexText-Konfigurationsdatei
- Art der eingefügten Komponente, z.B. "Textdatei" oder "Excel-Datei"

Wenn der Komponentename automatisch generiert wurde und Sie anschließend eine Instanzdatei auswählen, werden Sie [gefragt](#), ob Sie auch den Komponentennamen aktualisieren möchten.

Der Komponentename kann folgende Zeichen enthalten:

- Leerzeichen, z.B. "Text file" oder "Excel file". Vorangestellte oder nachgestellte



- Leerzeichen sind nicht zulässig.
- Punkte, z.B. Orders.EDI

Beachten Sie, dass einige Zeichen nur mit größten Schwierigkeiten oder gar nicht in die Befehlszeile eingegeben werden können und dass länderspezifische Zeichen in Windows und in der Befehlszeile unterschiedlich kodiert sein können.

Folgende Zeichen dürfen nicht in einem Komponentennamen vorkommen:

- Schrägstriche, umgekehrte Schrägstriche oder Doppelpunkte
- einfache oder doppelte Anführungszeichen
- die einzigen zulässigen Leerzeichen sind Whitespaces, d.h. Tabulatorzeichen oder CR/LF-Zeichen dürfen nicht verwendet werden.

**Komponenteneinstellungen**

Komponentenname:

Schema-Datei:

XML-Input-Datei:

XML-Output-Datei:

Target-Namespace-Präfix:

☒ Schema- / DTD-Referenz hinzufügen (leeres Feld für absoluten Pfad des Schemas):

☒ Werte in Zieltypen konvertieren (deaktivieren, um numerische oder Datumswerte beizubehalten, auf die Gefahr hin, dass eine ungültige Ausgabe erzeugt wird)

☒ Pretty Print für Ausgabe

☐ Digitale Signatur erstellen (nur Built-in-Ausführung)

Falls Signatur nicht erstellt werden kann: ☐ Verarbeitung stoppen ☒ Ohne Signatur fortfahren

Kodierung

Kodierungsname:

Bytefolge:  ☐ Bytefolge-Markierung inkl.

StyleVision Power Stylesheet-Datei:   

☒ Input-Verarbeitungsoptimierungen auf Basis von min/maxOccurs aktivieren

☒ Alle Dateipfade relativ zur MFD-Datei speichern

**Schema-Datei:** Zeigt den Dateinamen und Pfad des Zielschemas an.

**XML-Input-Datei:** Dient zum Auswählen oder Ändern der XML-Instanz für die aktive Schemakomponente. Dieses Feld wird ausgefüllt, wenn Sie die Schemakomponente zum ersten Mal einfügen und ihr eine XML-Instanzdatei zuweisen.

**XML-Output-Datei:** Dies ist der Datei- und Pfadname, unter dem die XML-Zielinstanz gespeichert wird, wenn Programmcode generiert und ausgeführt wird. Der Dateiname wird auch als erster Eintrag in der Komponente angezeigt.

Der Eintrag aus dem Feld XML-Input-Instanz wird automatisch in dieses Feld übernommen, wenn Sie die XML-Instanzdatei zuweisen. Wenn Sie der Komponente keine XML-Instanzdatei zuweisen, so enthält dieses Feld den Eintrag **schemafilenameandpath.xml**.

**Target-Namespace-Präfix:** Ermöglicht die Eingabe eines Präfixes für den Target-Namespace, wenn es sich um ein Schema / XML-Dokument handelt. Ein Target-Namespace muss im Zielschema definiert werden, damit man ihm hier das Präfix zuweisen kann.

**Schema- / DTD-Referenz hinzufügen:** Fügt den Pfad der referenzierten Schema-Datei zum Root-Element der XML-Ausgabe hinzu.

Wenn Sie einen Pfad in dieses Feld eingeben, können Sie definieren, wo sich die von der XML-Instanzdatei referenzierte Schemadatei befindet. Damit stellen Sie sicher, dass die Ausgabeinstanz im Mapping-Ordner bei Ausführung des Mappings validiert werden kann. Sie können in dieses Feld sowohl eine http:// Adresse als auch einen absoluten oder relativen Pfad eingeben.

Wenn Sie diese Option deaktivieren, können Sie die XML-Instanz vom referenzierten XML-Schema oder der DTD entkoppeln. Verwenden Sie diese Option z.B., wenn Sie die erzeugte XML-Ausgabedatei an jemanden senden möchten, der keinen Zugriff auf das zugrunde liegende XML-Schema hat.

**Zielwerte in Zieltypen konvertieren:** Damit können Sie festlegen, ob beim Mapping die XML-Zielschematypen verwendet werden sollen (Standardeinstellung - aktiviert), oder ob alle Daten, die auf die Zielkomponenten gemappt werden, als **Stringwerte** behandelt werden sollen.

Wenn Sie diese Option deaktivieren, können Sie die exakte Formatierung der Werte beibehalten, z.B. eignet sich die Option, wenn ein Inhalt genau einem Pattern Facet in einem Schema entsprechen muss, demzufolge ein numerischer Wert eine bestimmte Anzahl an Dezimalstellen aufweisen muss.

Sie können die Zahl mit Hilfe von Mapping-Funktionen als String im gewünschten Format formatieren und diesen String dann auf die Zielkomponente mappen.

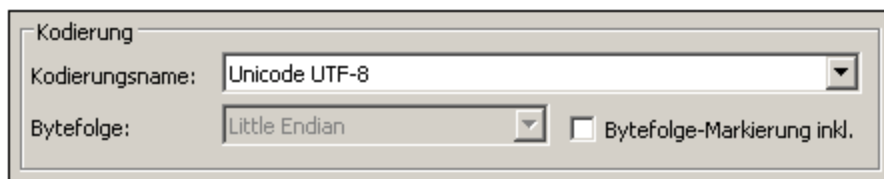
Beachten Sie: Wenn Sie diese Option deaktivieren, wird auch die Erkennung ungültiger Werte deaktiviert, z.B. das Schreiben von Buchstaben in numerische Felder.

**Pretty-Print für Ausgabe:** Formatiert Ihr XML-Dokument im Ausgabefenster neu, um eine strukturierte Anzeige des Dokuments zu generieren. (Jedes Subelement ist um einen Tabstopp vom übergeordneten Element eingerückt.)

**Digitale Signatur erstellen (nur Built-in-Ausführung):**

Digitale Signaturen können nur generiert werden, wenn das Ausgabeziel BUILTIN ist. Dies kann auch über die Befehlszeile mit dem Parameter /BUILTIN erfolgen. Nähere Informationen zu bestimmten Signatureinstellungen finden Sie unter [Erstellen einer digitalen Signatur](#).

## Kodierungseinstellungen



Ab MapForce Version 2008 hat jede Komponente, sowohl Quellen als auch Ziele, ihre eigenen Kodierungseinstellungen. Das bedeutet, dass die \*.mfd-Mapping-Dateien keine Standardkodierung haben. Jede Komponente, aus der sich das Mapping zusammensetzt hat ihre eigenen Kodierungseinstellungen. Bei Komponenten handelt es sich immer um XML-, Text-, EDI- und Flextext-Komponenten.

Es gibt jedoch eine Standardkodierungseinstellung, die unter **Extras | Optionen** auf dem Register "Allgemein" definiert ist. Sie heißt "Standardkodierung für neue Komponenten" und wird angewendet, wenn neue Komponenten erstellt/eingefügt werden. Wenn Mappings aus früheren Versionen geöffnet werden, wird die Standardkodierungseinstellung verwendet.

Die Gruppe Kodierung besteht aus drei Steuerelementen:

- Auswahlliste zur Auswahl des Kodierungsnamens.
- Auswahlliste zur Definition der Byte-Folge (little endian, big endian).
- Kontrollkästchen "Bytefolge-Markierung inkludieren".

Die Standardeinstellungen sind:

- UTF-8
- little endian (deaktiviert bei UTF-8)
- keine Bytefolge-Markierung.

Bitte beachten Sie:

Die Aktivierung des Kontrollkästchens "Bytefolge-Markierung inkludieren" im Dialogfeld "Komponenteneinstellungen" hat keinerlei Auswirkung auf die Ausgabe von **XSLT 1.0/2.0**, da diese Sprachen BOMs (Byte Order Marks) nicht unterstützen.

#### **Input-Verarbeitungsoptimierungen auf Basis von min/maxOccurs aktivieren**

In MapForce Version 2009 gibt es nun eine spezielle Behandlung für Sequenzen, von denen bekannt ist, dass sie nur genau ein Datenelement enthalten, z.B. erforderliche Attribute oder Child-Elemente mit minOccurs und maxOccurs = 1. In diesem Fall wird das erste Datenelement der Sequenz extrahiert, anschließend wird das Datenelement direkt als atomarer Wert (und nicht als Sequenz) verarbeitet.

Wenn die Input-Daten gemäß dem Schema nicht gültig sind, könnte eine leere Sequenz in einem Mapping vorkommen, sodass das Mapping mit einer Fehlermeldung abgebrochen wird. Damit auch ein solcher **ungültiger Input** verarbeitet werden kann, kann diese Optimierung in den Komponenteneinstellungen von XML- und EDI-Komponenten deaktiviert werden.

Die Datenbankeinstellungen für das folgende Dialogfeld werden nur angezeigt, wenn Sie das Dialogfeld "Komponenteneinstellungen" einer Datenbankkomponente öffnen.

**Komponenteneinstellungen**

**Datenbank**

Datenquelle: 2012\MapForceExamples\AltovaTarget.mdb Ändern

Verbindungsname: AltovaTarget

Datenbankart: MS Access ( ADO )

Connection String: Data Source=C:\Documents and Settings\My\My Documents\Altova\MapForce2012\MapForceExamples\AltovaTarget.mdb;Provider=Microsoft.Jet.OLEDB.4.0

**Login-Einstellungen**

Benutzer:

Passwort:

**JDBC-spezifische Einstellungen**

JDBC-Treiber: sun.jdbc.odbc.JdbcOdbcDriver

Datenbank-URL: jdbc:odbc;;DRIVER=Microsoft Access Driver (\*.mdb);DBQ=

**ADO/OLEDB-spezifische Einstellungen**

Datenquelle: tova\MapForce2012\MapForceExamples\AltovaTarget.mdb

Katalog: AltovaTarget

Provider: Microsoft.Jet.OLEDB.4.0

Zus. Optionen:

**Generierungseinstellungen**

☐ Transaktionen verwenden

☐ Schema-Namen aus Tabellennamen entfernen

OK Abbrechen

### Datenbank

**Datenquelle:** Zeigt den Namen der aktuellen Datenbankkomponente an. Wenn Sie auf die Schaltfläche **Ändern** klicken, können Sie eine andere Datenbank auswählen oder die Tabellen, die in der Komponente enthalten sein sollen, neu definieren, wenn Sie dieselbe Datenbank auswählen. Die Konnektoren zu den Tabellen desselben Namens werden beibehalten.

Sie können die Tabellen in der Komponente auch ändern, indem Sie mit der rechten Maustaste auf eine Datenbankkomponente klicken und den Befehl **Tabellen hinzufügen/entfernen** wählen.

**Connection String:** Zeigt den aktuellen Datenbank Connection String an. Dieses Feld kann nicht bearbeitet werden.

### Login-Einstellungen

**Datenqu.:** Zeigt den Namen der Datenquelle an.

**Katalog:** Zeigt den Namen der spezifischen Datenbank an.

**Benutzer:** Geben Sie, falls erforderlich, den für den Zugriff auf die Datenbank erforderlichen Benutzernamen ein.

**Passwort:** Geben Sie, falls erforderlich, das für den Zugriff auf die Datenbank erforderliche Passwort ein.

Die Login-Einstellungen werden für alle Codegenerierungsziele und den Built-in-Ausführungsprozessor verwendet.

### JDBC-spezifische Einstellungen

**JDBC-Treiber:** Zeigt den derzeit aktiven Treiber für die Datenbankkomponente an. Beim Definieren der Datenbankkomponente wird automatisch der Standardtreiber übernommen. Sie können den hier definierten Treiber Ihren Bedürfnissen entsprechend ändern. Achten Sie darauf, dass die Syntax des Eintrags im Feld "Datenbank-URL" mit dem von Ihnen gewählten Treiber konform ist.

Die JDBC-spezifischen Einstellungen werden nur für die Java-Codegenerierung verwendet.

**Datenbank-URL:** URL der aktuell ausgewählten Datenbank. Stellen Sie sicher, dass dieser Eintrag mit der JDBC-Treibersyntax des jeweils im Feld JDBC-Treiber definierten Treibers konform ist.

**Benutzer:** Geben Sie hier ggf. den Benutzernamen ein, über den Sie auf die Datenbank zugreifen.

**Passwort:** Geben Sie hier ggf. das Passwort ein, um auf die Datenbank zugreifen zu können.

### ADO/OLEDB-spezifische Einstellungen:

**Provider:** Zeigt den gerade aktiven Provider für die Datenbankkomponente an. Der Provider wird beim Definieren der Datenbankkomponente automatisch eingetragen.

Die ADO/OLE-Datenbankeinstellungen werden nur für die C++ und C#-Codegenerierung verwendet. Die Einstellungen der Datenquelle und Kataloge werden für den Built-in-Ausführungsprozessor nicht verwendet.

**Zus.. Optionen:** Zeigt zusätzliche Datenbankoptionen an.

### Generierungseinstellungen:

**Schemanamen aus Tabellennamen entfernen:** Damit können Sie Datenbankschemanamen aus generiertem Code entfernen, sodass aus Gründen zusätzlicher Flexibilität nur die Tabellennamen beibehalten werden.

Beachten Sie, dass diese Option bei SQL Select-Anweisungen nicht funktioniert.

**Transaktionen verwenden:** Aktiviert den [transaktionsgesteuerten Datenbankzugriff](#), wenn als Ziel eine Datenbank verwendet wird. Wenn ein Fehler auftritt, wird ein Dialogfeld geöffnet, in dem Sie die weitere Vorgehensweise festlegen können. Bei Auswahl dieser Option wird die Transaktionssteuerung für alle Tabellen der Datenbankkomponente aktiviert.

Die Einstellungen für die Codegenerierung gelten für alle Zielsprachen sowie für den Built-in-Ausführungsprozessor.

## 30.6 Verbindung

### Idente Sub-Einträge automatisch verbinden

Aktiviert bzw. deaktiviert die Funktion "Idente Sub-Einträge automatisch verbinden" sowie das dazugehörige Symbol in der Symbolleiste.

### Einstellungen für "Idente Sub-Einträge verbinden"

Öffnet das Dialogfeld "Einstellungen für 'Idente Sub-Einträge verbinden'", in dem Sie die Verbindungseinstellungen definieren können.

### Idente Sub-Einträge verbinden

Mit diesem Befehl können Sie sowohl im Quell- als auch im Zielschema mehrere Konnektoren für Datenelemente **desselben** Namens erzeugen. Die Einstellungen, die Sie in diesem Dialogfeld definieren, werden beibehalten und beim Verbinden zweier Elemente angewendet,

wenn in der Titelleiste das Symbol "Idente Sub-Einträge automatisch verbinden"  aktiv ist. Durch Klicken auf das Symbol können Sie es aktivieren bzw. deaktivieren. Nähere Informationen dazu finden Sie im Abschnitt "[Konnektoreigenschaften](#)".

### Zielorientiert (Standard)

Ändert den Konnektortyp in Standard-Mapping. Nähere Informationen siehe [Quellorientiertes/Mixed Content und Standard-Mapping](#).

### Alles kopieren (Sub-Einträge kopieren)

Erstellt Konnektoren für alle übereinstimmenden Sub-Einträge, wobei jeder der Child-Konnektoren als Substruktur des Parent-Konnektors dargestellt wird. Nähere Informationen siehe ["Alles kopieren"-Verbindungen](#)

### Quellorientiert (Mixed Content)

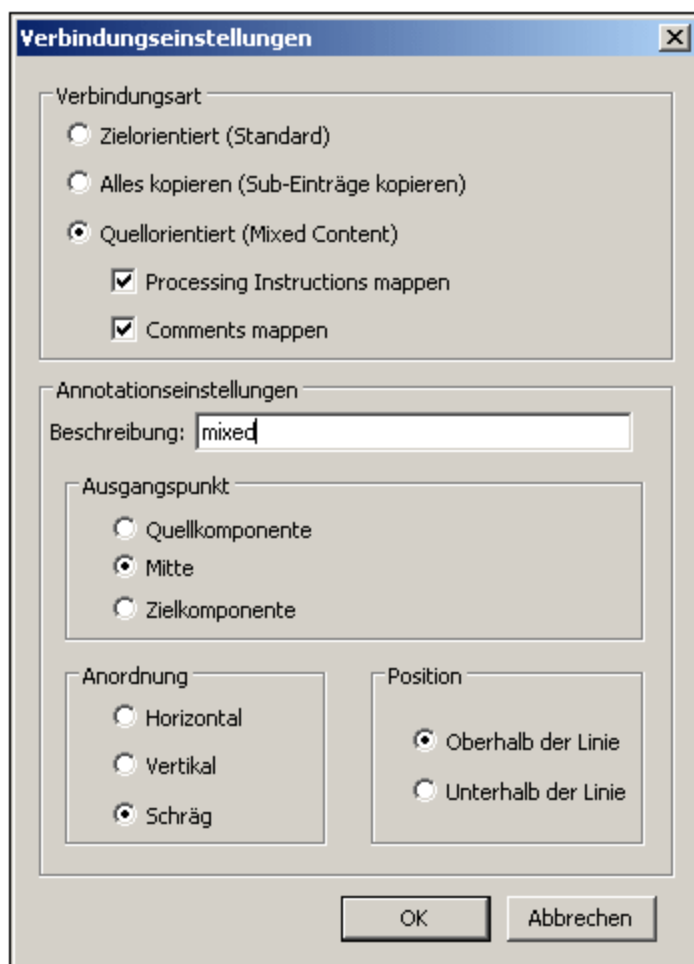
Ändert den Konnektortyp in quellorientiert / mixed content und gestattet die Auswahl zusätzlicher Elemente für das Mapping. Die zusätzlichen Elemente müssen **Child-Elemente** des gemappten Datenelements in der XML-Quelldatei sein, damit sie gemappt werden können. Nähere Informationen siehe [Standardeinstellungen: Mixed Content](#).

### Eigenschaften:

Öffnet ein Dialogfeld, in dem Sie die jeweiligen (mixed content) Einstellungen für den aktuellen Konnektor definieren können. Beachten Sie, dass Optionen, die nicht zur Verfügung stehen, ausgegraut sind.

Bitte beachten Sie, dass diese Einstellungen auch für **complexType**-Elemente gelten, die keine Text-Nodes enthalten!



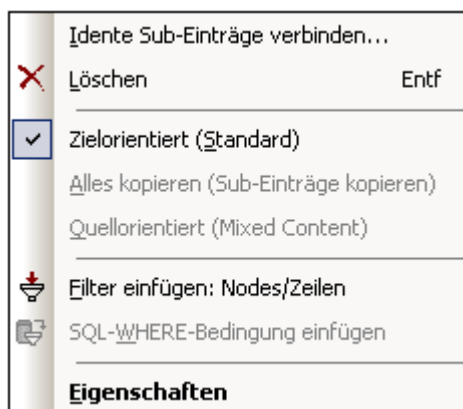


### Annotationseinstellungen:

Einzelne Konnektoren können zur besseren Unterscheidung beschriftet werden.

1. Doppelklicken Sie auf einen Konnektor und geben Sie den Namen des Konnektors in das Feld "Beschreibung" ein. Dadurch werden alle Optionen im Bereich "Annotationseinstellungen" aktiv.
2. In den restlichen Bereichen können Sie die Position und Ausrichtung der Beschriftung definieren.

### Konnektor-Kontextmenü:



**Idente Sub-Einträge verbinden**

Öffnet das Dialogfeld "Idente Sub-Einträge verbinden", in dem Sie die Verbindungseinstellungen ändern und die Datenelemente verbinden können, wenn Sie auf OK klicken.

**Löschen**

Löscht den markierten Konnektor.

**Zielorientiert (Standard)**

Ändert den Konnektortyp in Standard-Mapping, Nähere Informationen dazu siehe: "[Quellorientiertes/Mixed Content und Standard-Mapping](#)".

**Alles kopieren (Sub-Einträge kopieren)**

Ändert den Konnektortyp in "Alles kopieren" und verbindet alle Sub-Einträge desselben Namens auf eine grafisch optimierte Weise. Nähere Informationen dazu siehe "[Alles kopieren Konnektoren](#)".

**Quellorientiert (Mixed Content)**

Ändert den Konnektortyp in "Quellorientiert / Mixed Content". Nähere Informationen dazu siehe: "[Quellorientiertes und Mixed Content Mapping](#)".

**Filter einfügen: Nodes/Zeilen**

Fügt eine Filterkomponente in den Konnektor ein. Der Quellkonnektor wird mit dem Parameter "nodes/row" verbunden, der Zielkonnektor mit dem "on-true" Parameter. Nähere Informationen dazu finden Sie unter [Filtern - Abrufen dynamischer Daten, Lookup-Tabelle](#).

**SQL-Where-Bedingung einfügen**

Fügt eine SQL-Where-Komponente in den Konnektor ein. Der Quellkonnektor wird mit dem table-Parameter verbunden, der Zielkonnektor mit dem result-Parameter. Nähere Informationen dazu finden Sie unter [SQL SELECT-Anweisungen als virtuelle Tabellen](#).

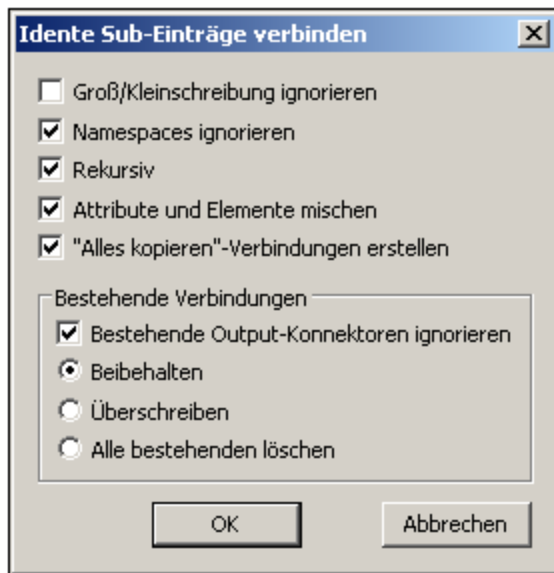
**Eigenschaften**

Öffnet das Dialogfeld "Verbindungseinstellungen", in dem Sie die jeweiligen mixed content-Einstellungen sowie die Einstellungen für Konnektor-Annotationen vornehmen können. Nähere Informationen dazu finden Sie im Referenzteil im Abschnitt [Verbindung](#).

**Dialogfeld "Idente Sub-Einträge verbinden"**

Mit diesem Befehl können Sie mehrere Konnektoren zwischen Datenelementen erstellen, die in der Quell- und in der Zielkomponente **denselben Namen** haben.

1. Verbinden Sie zwei übergeordnete Datenelemente, die in beiden Komponenten **Child-Elemente** desselben Namens aufweisen.
2. Rechtsklicken Sie auf den Konnektor und wählen Sie die Option **Idente Sub-Einträge verbinden**.



3. Wählen Sie die weiter unten beschriebenen erforderlichen Optionen aus und klicken Sie auf OK, um das Mapping durchzuführen.

Für alle Child-Elemente, die denselben Namen haben und den im Dialogfeld definierten Einstellungen entsprechen, werden Mappings erzeugt.

Bitte beachten Sie:

Die hier definierten Einstellungen werden beibehalten und angewendet, wenn in der Titelleiste das Symbol "**Identische Sub-Einträge automatisch verbinden**"  aktiv ist. Durch Klicken auf das Symbol können Sie es aktivieren bzw. deaktivieren.

**Groß/Kleinschreibung ignorieren:**

Ignoriert die Groß- und Kleinschreibung der Child-Elemente.

**Namespaces ignorieren:**

Ignoriert die Namespaces der Child-Elemente.

**Rekursiv:**

Nachdem das erste Konnektoren-Set erzeugt wurde, werden auch die Elemente der nächsten untergeordneten Ebene auf identische Namen hin überprüft und ggf. auch für diese Konnektoren erstellt. Nun werden die Child-Elemente dieser untergeordneten Elemente überprüft usw.

**Attribute und Elemente mischen:**

Damit können Konnektoren zwischen Datenelementen desselben Namens erzeugt werden, selbst wenn es sich um verschiedene Typen handelt, z.B. wenn zwei Datenelemente "Name" existieren, eines jedoch ein Element und das andere ein Attribut ist. Ist die Option aktiviert, wird ein Konnektor zwischen diesen Datenelementen erzeugt.

**Bestehende Verbindungen:**

**Bestehende Ausgabe-Verbindung ignorieren:**

Erzeugt **zusätzliche** Konnektoren zu anderen Komponenten, selbst wenn die derzeit vorhandenen Output-Symbole bereits Konnektoren haben.

**Beibehalten**

Behält bestehende Konnektoren bei.

**Überschreiben:**

Erzeugt Konnektoren gemäß den definierten Einstellungen neu. Bestehende Konnektoren

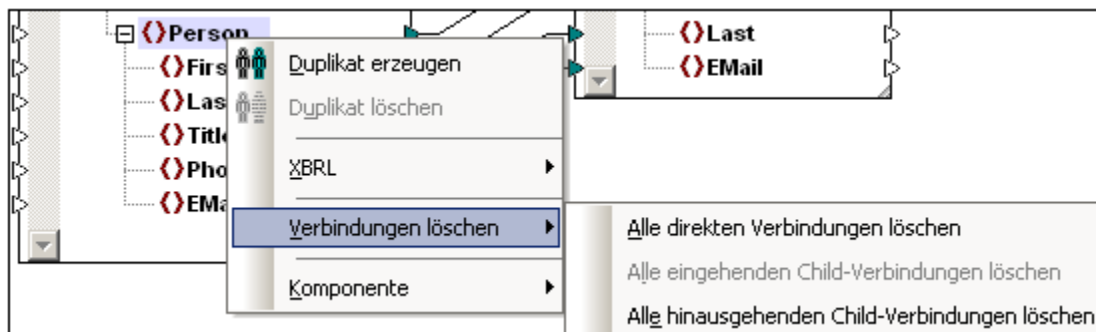
werden gelöscht.

#### **Alle bestehenden löschen:**

Löscht alle bestehenden Konnektoren, bevor neue erzeugt werden.

#### **Löschen von Verbindungen**

Konnektoren, die über das Dialogfeld "Identische Sub-Einträge verbinden" oder während des Mappings erstellt wurden, können als Gruppe entfernt werden.



Rechtsklicken Sie auf den Datenelementnamen der Komponente, nicht den Konnektor selbst - in diesem Beispiel auf "Person".

Wählen Sie den Befehl **Verbindungen löschen | Alle... Verbindungen löschen**.

#### **Alle direkten Verbindungen löschen:**

Löscht alle Konnektoren, die direkt auf eine andere Quelle- oder Zielkomponente gemappt sind, bzw. auf die eine andere Zielkomponente gemappt ist.

#### **Alle eingehenden Child-Verbindungen löschen:**

Ist nur aktiv, wenn Sie mit der rechten Maustaste auf ein Datenelement in einer Zielkomponente geklickt haben. Löscht alle eingehenden Child-Konnektoren.

#### **Alle hinausgehenden Child-Verbindungen löschen:**

Ist nur aktiv, wenn Sie mit der rechten Maustaste auf ein Datenelement in einer Quellkomponente geklickt haben. Löscht alle hinausgehenden Child-Konnektoren.

## 30.7 Funktion

### Benutzerdefinierte Funktion erstellen...

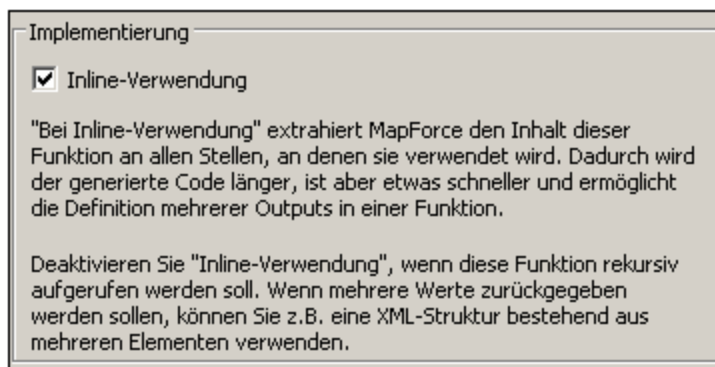
Erstellt eine neue benutzerdefinierte Funktion. Bei Auswahl dieser Option wird eine leere benutzerdefinierte Funktion erstellt, in die Sie die benötigten Komponenten einfügen können. Bei Definition einer solchen Funktion wird automatisch eine einzige Ausgabekomponente eingefügt. Eine benutzerdefinierte Funktion kann nur eine Output-Komponente haben, es sei denn, sie ist als inline-Funktion definiert. Nähere Informationen dazu finden Sie unter "[So erstellen Sie eine benutzerdefinierte Funktion von Grund auf neu](#)".

### Benutzerdefinierte Funktion von Auswahl erstellen:

Erstellt auf Basis der aktuell im Projektfenster ausgewählten Elemente eine neue benutzerdefinierte Funktion. Nähere Informationen dazu finden Sie unter "[Hinzufügen benutzerdefinierte Funktionen](#)".

### Funktion - Einstellungen

Öffnet das Einstellungsdialogfeld für die derzeit aktive benutzerdefinierte Funktion, wo Sie die aktuellen Einstellungen ändern können. Auf diese Art können Sie den Typ der benutzerdefinierten Funktion ändern, d.h. doppelklicken Sie auf die Titelleiste einer benutzerdefinierten Funktion, um den Inhalt zu sehen und wählen Sie anschließend diese Menüoption, um den Typ zu ändern.



### Funktion entfernen

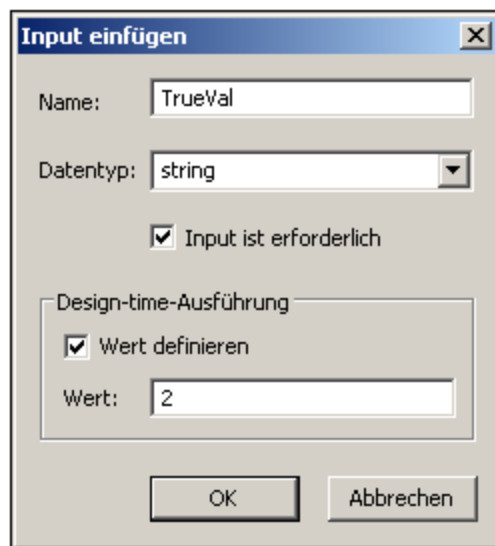
Löscht die aktuell aktive benutzerdefinierte Funktion vom Register dieses Namens, während Sie an bestehenden benutzerdefinierten Funktionen arbeiten. D.h. dies funktioniert nur bei bestehenden benutzerdefinierten Funktionen, wenn Sie deren Inhalt anzeigen.

Es erscheint eine Meldung, die Sie daran erinnert, dass Instanzen ungültig werden können. Sie listet die Bibliotheken auf, in denen die benutzerdefinierte Funktion vorkommt.

### Input-Komponente einfügen

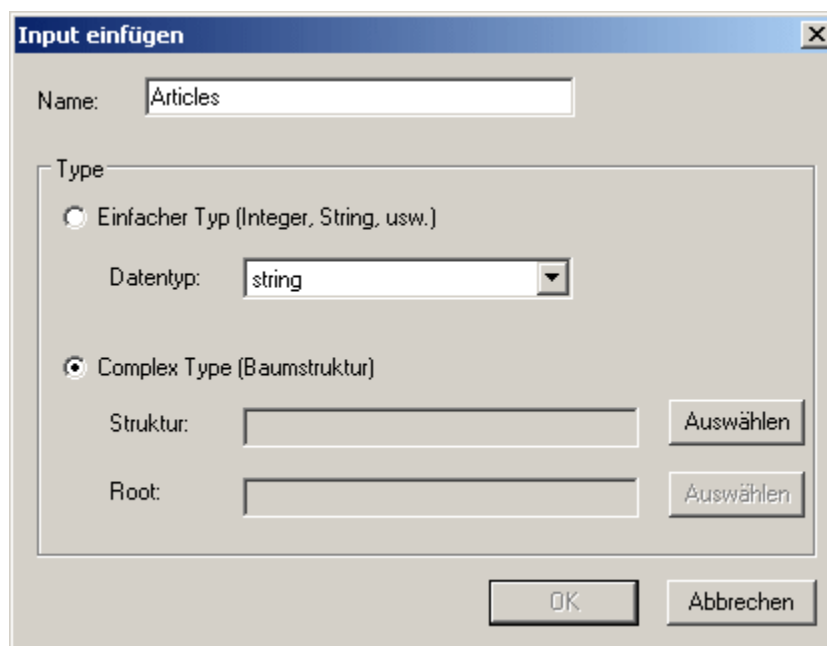
Fügt eine "Input"-Komponente in das Mapping oder in eine benutzerdefinierte Funktion ein.

Wenn Sie sich auf der Registerkarte für das Haupt-Mapping befinden, wird das unten abgebildete Dialogfeld angezeigt. Bei dieser Art von Input-Komponente können Sie bei der Ausführung des kompilierten Mappings über die Befehlszeile einen **Parameter** definieren. Nähere Informationen dazu finden Sie unter "[Befehlszeile - Definieren von Input/Output-Dateien](#)".



Wenn Sie sich auf dem Register für die benutzerdefinierte Funktion befinden, wird das unten gezeigte Dialogfeld angezeigt. Dieser Typ der Input-Komponente gestattet Ihnen:

- einfache Inputs zu definieren
- komplexe Inputs zu definieren z.B. Schema-Strukturen



### Output-Komponente einfügen

Fügt eine "Output"-Komponente in eine benutzerdefinierte Funktion ein. Auf einem benutzerdefinierten Funktionsregister wird das unten gezeigte Dialogfeld angezeigt. Mit diesem Typ der Input-Komponente können Sie Folgendes definieren:

- einfache Outputs
- komplexe Outputs z.B. Schema-Strukturen

**Output einfügen**

Name:

Typ

☐ Einfacher Typ (Integer, String, usw.)

Datentyp:

☒ Complex Type (Baumstruktur)

Struktur:

Root:

☒ Strukturgepfad relativ zur MFD-Datei speichern

☒ Output ist eine Sequenz

## 30.8 Ausgabe

Mit der ersten Gruppe von Optionen (**XSLT 1.0**, **XSLT 2.0**, **usw.**) können Sie die Zielsprache definieren, in der Ihr Code erzeugt werden soll.

Beachten Sie, dass der Built-In-Ausführungsprozessor eine Vorschau auf das Ergebnis des Mappings bietet, wenn Sie auf die Schaltfläche "Ausgabe" klicken, und nicht zum Generieren von Programmcode verwendet werden kann. Nähere Informationen dazu finden Sie im Abschnitt "[Built-In-Ausführungsprozessor](#)".

### **Ausgabedatei validieren**

Validiert die erzeugte XML-Datei anhand des referenzierten Schemas.

### **Generierte Ausgabedatei speichern**

Speichert die aktuell auf dem Register "Ausgabe" angezeigten Daten.

### **Alle Ausgabedateien speichern**

Speichert alle von dynamischen Mappings generierten Ausgabedateien. Nähere Informationen dazu finden Sie unter [Dynamische Dateinamen - Input / Output](#).

### **Ausgabedatei neu generieren**

Generiert das aktuelle Mapping vom Ausgabefenster aus neu. Wenn im Ausgabefenster gerade ein SQL-Skript angezeigt wird, führt das Skript das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.

### **SQL-Skript ausführen**

Führt das Mapping auf die Zieldatenbank unter Berücksichtigung der definierten Tabellenaktionen aus.

### **Lesezeichen einfügen/löschen**

Fügt im Fenster Ausgabe an der Cursorposition ein Lesezeichen ein.

### **Nächstes Lesezeichen**

Navigiert im Fenster Ausgabe zum nächsten Lesezeichen.

### **Vorhergehendes Lesezeichen**

Navigiert im Fenster Ausgabe zum vorhergehenden Lesezeichen.

### **Alle Lesezeichen löschen**

Entfernt im Fenster Ausgabe alle derzeit definierten Lesezeichen.

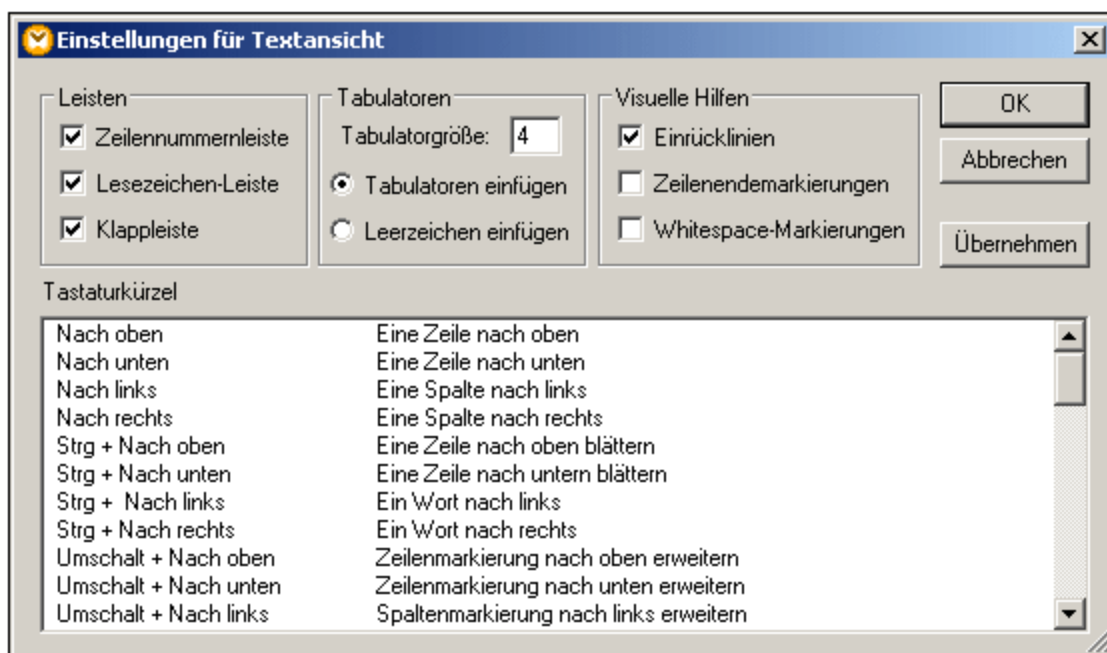
### **Pretty-Print**

Formatiert Ihr XML-Dokument im Fenster Ausgabe neu, sodass Sie eine strukturierte Übersicht über das Dokument haben. Die einzelnen Child Nodes werden durch ein einziges Tabulatorzeichen vom übergeordneten Node eingerückt.

### **Einstellungen für Textansicht**

Dient zum Anpassen der Text Einstellungen im Fenster Ausgabe und zeigt die aktuell definierten Tastaturkürzel an, die im Fenster gelten.





## 30.9 Ansicht

### Annotationen anzeigen

Zeigt XML-Schema-Annotationen (sowie EDI-Informationen) im Komponentenfenster an. Wenn auch das Symbol "Datentypen anzeigen" aktiv ist, werden beide Informationen in Tabellenform angezeigt.

|         |                     |
|---------|---------------------|
| = F1060 |                     |
| type    | string              |
| ann.    | Revision identifier |

### Datentypen anzeigen

Zeigt die Schema-Datentypen für jedes Element/Attribut an. Wenn auch das Symbol "Annotationen anzeigen" aktiv ist, werden beide Informationen in Tabellenform angezeigt.

### Bibliothek in Titelleiste anzeigen

Zeigt den Namen der Bibliothek im Funktionstitel an.

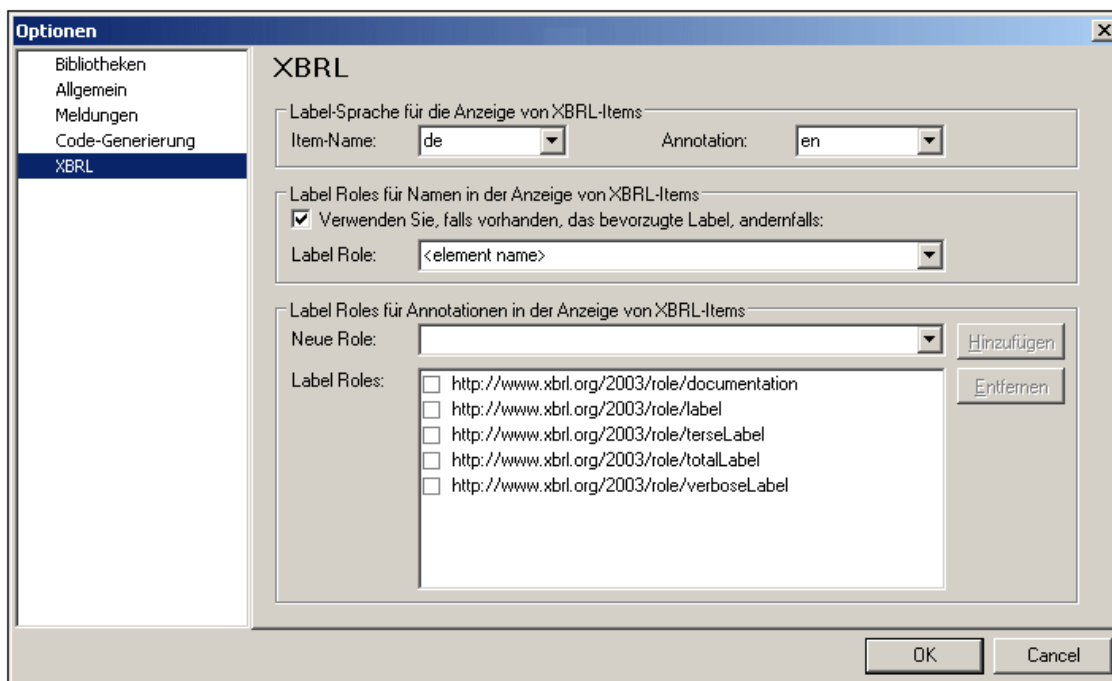
### Tipps anzeigen

Zeigt einen Tooltip mit erklärendem Text an, wenn Sie den Mauszeiger über einer Funktion platzieren.

### XBRL-Anzeigeoptionen

Dient zum Definieren bestimmter Anzeigeeinstellungen für XBRL-Komponenten.

- Die Label-Sprache für die XBRL Items und ihre Annotationen.
- Die bevorzugten Label-Rollen für XBRL Item-Namen
- Den spezifischen Typ von Label-Rollen von Annotationen für XBRL Items



**Ausgewählte Komponenten-Konnektoren anzeigen** 

Wechselt zwischen zwei Ansichten:

- Anzeige aller Mapping-Konnektoren oder
- Anzeige nur jener Konnektoren, die sich auf die aktuell ausgewählten Komponenten beziehen.

**Quell- und Zielkonnektoren anzeigen** 

Wechselt zwischen zwei Ansichten:

- Anzeige von Konnektoren, die mit der aktuell ausgewählten Komponente **direkt** verbunden sind oder
- Anzeige von Konnektoren, die mit der aktuell ausgewählten Komponente verknüpft sind und von der Quellkomponente ausgehen und zur Zielkomponente führen.

**Vergrößern/Verkleinern**

Öffnet das Dialogfeld "Vergrößern/Verkleinern", in dem Sie entweder einen numerischen Zoom-Faktor eingeben oder den Zoom-Faktor mit Hilfe des Schiebereglers interaktiv einstellen können.

**Zurück**

Geht auf den Registern der gerade geöffneten Mappings zurück.

**Vorwärts**

Geht auf den Registern der gerade geöffneten Mappings vorwärts.

**Statusleiste**

Blendet die unterhalb des Fensters "Meldungen" angezeigte Statusleiste ein oder aus.

**Bibliotheksfenster**

Blendet das Bibliotheksfenster, das alle Bibliotheksfunktionen enthält, ein oder aus.

**Meldungen**

Blendet das Fenster "Meldungen" ein oder aus. Bei Generierung von Code wird das Validierungsfenster automatisch aktiviert, um das Ergebnis der Validierung anzuzeigen.

**Übersichtsfenster**

Blendet das Übersichtsfenster ein oder aus. Ziehen Sie das Rechteck mit der Maus, um in Ihrer Mapping-Projektansicht zu navigieren.

**Projektfenster**

Blendet das Projektfenster ein oder aus.

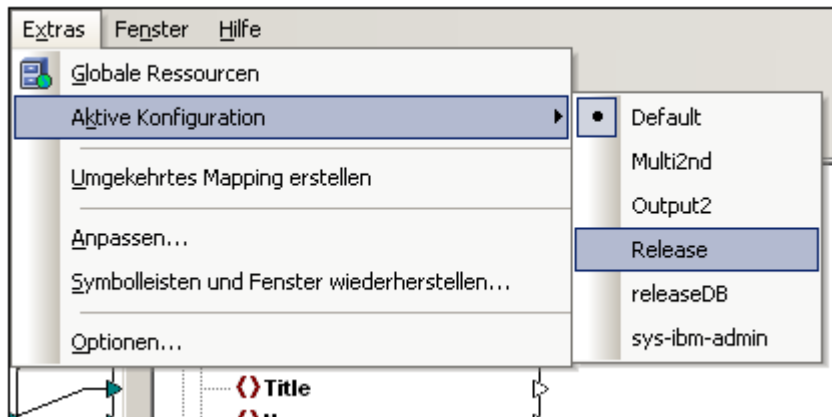
## 30.10 Extras

### Globale Ressourcen

Öffnet das Dialogfeld "Globale Ressourcen verwalten", in dem Sie globale Ressourcen zur XML-Datei für globale Ressourcen hinzufügen, bearbeiten und daraus löschen können. Nähere Informationen finden Sie unter [Globale Ressourcen - Eigenschaften](#).

### Aktive Konfiguration

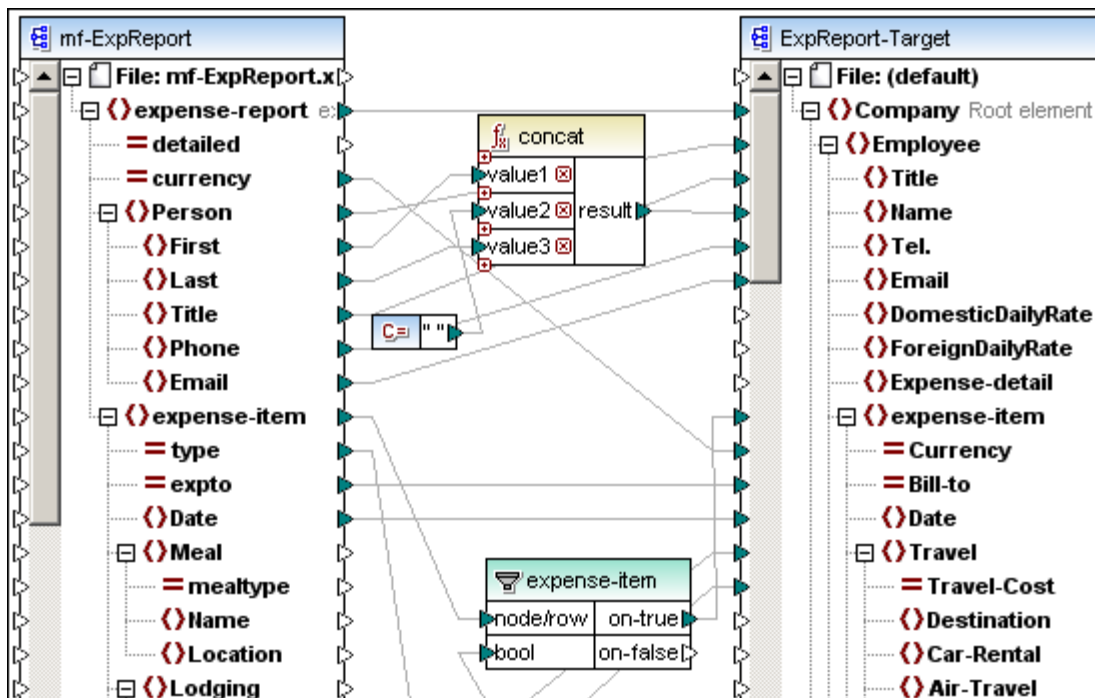
Dient zum Auswählen/Wechseln der aktuell aktiven globalen Ressource aus einer Liste aller Ressourcen/Konfigurationen in den globalen Ressourcen. Wählen Sie die gewünschte Konfiguration im Untermenü aus.



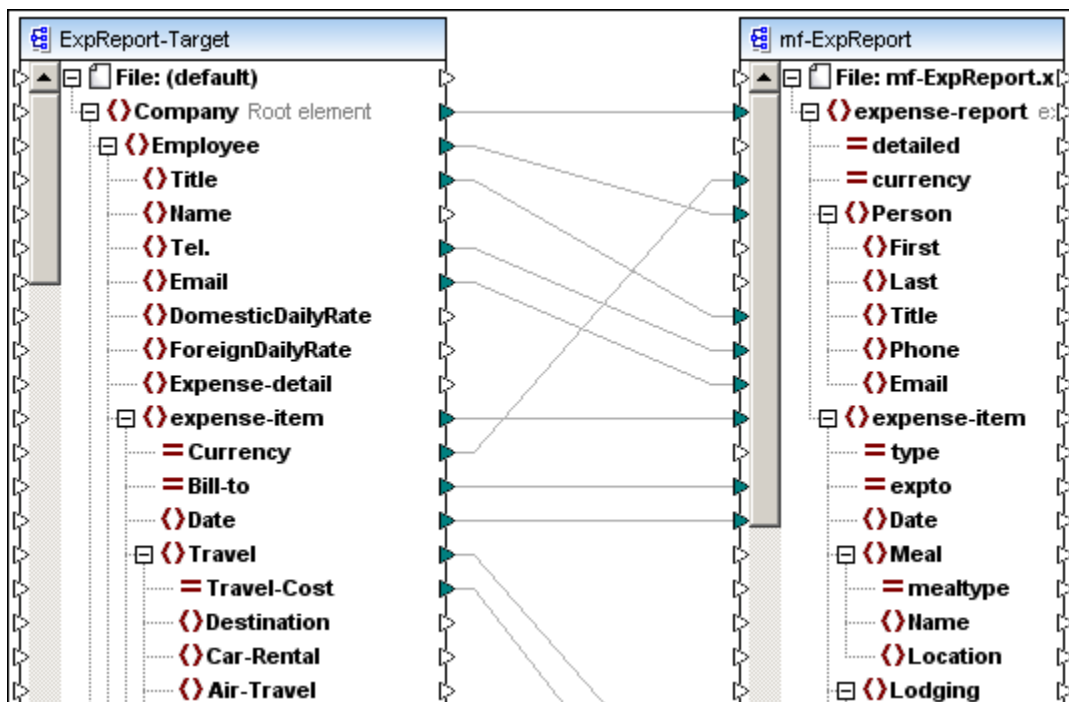
### Umgekehrtes Mapping erstellen

Erstellt anhand des gerade in MapForce aktiven Mappings, das die Basis für ein neues Mapping bilden soll, ein "umgekehrtes" Mapping. Beachten Sie, dass das Ergebnis nicht als vollständiges Mapping gedacht ist. Nur die direkten Verbindungen zwischen Komponenten werden im umgekehrten Mapping beibehalten. Das Mapping wird höchstwahrscheinlich nicht gültig sein oder kann nicht ausgeführt werden, wenn Sie auf die Schaltfläche "Ausgabe" klicken, ohne das Mapping vorher manuell zu bearbeiten.

Beispiel: **Tut-ExpReport.mfd** in Ordner ...MapForceExamples\Tutorial:



Ergebnis eines umgekehrten Mappings:



Allgemein:

- Die Quellkomponente wird zur Zielkomponente und die Zielkomponente wird zur Quellkomponente.
- Wenn der Komponente eine XML-Input- und eine XML-Output-Instanzdatei zugewiesen wurde, so werden die beiden gegeneinander ausgetauscht.

**Beibehaltene Verbindungen**

- direkte Verbindungen zwischen Komponenten.
- direkte Verbindungen zwischen Komponenten in einem verketteten Mapping
- die Art der Verbindung: Standard, gemischter Inhalt, Alles kopieren
- Einstellungen für die Weiterleitung einer Komponente
- Datenbankkomponenten bleiben unverändert

**Gelöschte Verbindungen**

- Verbindungen über Funktionen, Filter werden zusammen mit der Funktion gelöscht
- benutzerdefinierte Funktionen
- Webservice-Komponenten

**Symbolleisten und Fenster wiederherstellen**

Setzt die Symbolleisten, Eingabehilfenfenster, angedockten Fenster usw. wieder auf ihre Standardeinstellung zurück. Damit die Änderungen wirksam werden, muss MapForce neu gestartet werden.

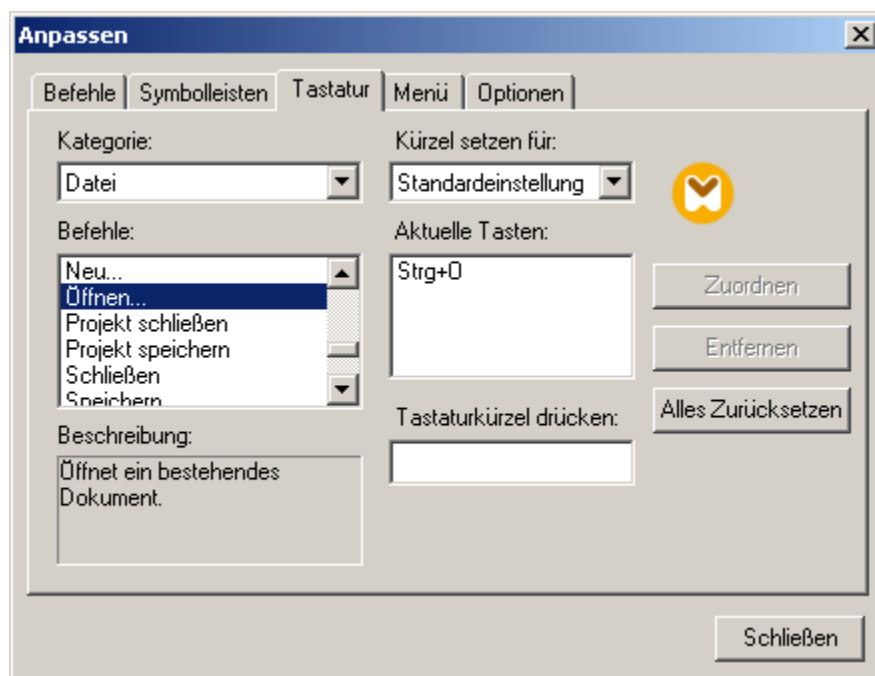
**Anpassen...**

Mit Hilfe des Befehls "Anpassen" können Sie MapForce Ihren Bedürfnissen entsprechend anpassen.

Auf dem Register **Tastatur** können Sie Tastenkürzel für jeden beliebigen MapForce Befehl definieren oder ändern.

**So weisen Sie einem Befehl ein neues Tastenkürzel zu:**

1. Wählen Sie den Befehl **Extras | Anpassen** und klicken Sie auf das Register "Tastatur".
2. Klicken Sie auf die Auswahlliste **Kategorie**, um den Menünamen auszuwählen.
3. Wählen Sie in der Liste **Befehle** den Befehl aus, dem Sie ein neues Tastenkürzel zuweisen möchten.
4. Klicken Sie in das Textfeld **Tastaturkürzel drücken** und drücken Sie die Tasten, mit denen der Befehl aufgerufen werden soll.



Die Tasten werden sofort im Textfeld angezeigt. Wenn das Tastaturkürzel bereits vergeben ist, wird diese Funktion unterhalb des Textfelds angezeigt.

5. Klicken Sie auf die Schaltfläche **Zuweisen**, um das Tastaturkürzel zuzuweisen. Das Tastaturkürzel wird nur im Listefeld "Aktuelle Tasten" angezeigt. (Um den Eintrag im Textfeld "Tastaturkürzel drücken" zu löschen, drücken Sie eine der Steuerungstasten: **Strg**, **ALT** oder **Umschalt**).

#### So heben Sie die Zuweisung eines Tastaturkürzels auf oder löschen es:

1. Klicken Sie im Listefeld "Aktuelle Tasten" auf das gewünschte Tastaturkürzel.
2. Klicken Sie auf die Schaltfläche **Entfernen**.
3. Klicken Sie zur Bestätigung auf die Schaltfläche **Schließen**.

#### Kürzel setzen für:

Derzeit keine Funktion.

#### Liste der aktuellen Tastaturkürzel-Zuweisungen:

##### Tastaturkürzel nach Taste

|               |                                               |
|---------------|-----------------------------------------------|
| F1            | Menü "Hilfe"                                  |
| F2            | Nächstes Lesezeichen (im Fenster "Ausgabe")   |
| F3            | Weitersuchen                                  |
| F10           | Menüleiste aktivieren                         |
| Num +         | Aktuellen Node erweitern                      |
| Num -         | Node reduzieren                               |
| Num *         | Alle unterhalb des aktuellen Node erweitern   |
| Strg + TAB    | Wechselt zwischen Mappings                    |
| Strg + F6     | Wechselt zwischen offenen Fenstern            |
| Strg + F4     | Schließt das aktive Mapping-Dokument          |
| Alt + F4      | Schließt MapForce                             |
| Alt + F, F, 1 | Öffnet die letzte Datei                       |
| Alt + F, T, 1 | Öffnet das letzte Projekt                     |
| Strg + N      | Datei neu                                     |
| Strg + O      | Datei öffnen                                  |
| Strg + S      | Datei speichern                               |
| Strg + P      | Datei drucken                                 |
| Strg + A      | Alles auswählen                               |
| Strg + X      | Ausschneiden                                  |
| Strg + C      | Kopieren                                      |
| Strg + V      | Einfügen                                      |
| Strg + Z      | Rückgängig                                    |
| Strg + Y      | Wiederherstellen                              |
| Entf          | Komponente löschen (mit Betätigungsmeldung)   |
| Umschalt+Entf | Komponente löschen (ohne Bestätigungsmeldung) |
| Strg + F      | Suchen                                        |
| F3            | Weitersuchen                                  |
| Umschalt+F3   | Vorheriges suchen                             |

**Pfeiltasten**

(nach oben / nach unten)

nächstes Datenelement der Komponente auswählen

Esc.

Bearbeitungen verwerfen / Dialogfeld schließen

Eingabetaste

Bestätigt eine Auswahl

**Tastaturkürzel im Fenster****"Ausgabe"**

Strg + F2

Lesezeichen einfügen/löschen

F2

Nächstes Lesezeichen

Umschalt + F2

Vorhergehendes Lesezeichen

Strg + Umschalt + F2

Alle Lesezeichen löschen

**Tastaturkürzel zum Vergrößern/Verkleinern**

Strg + Mausrad vorwärts

Vergrößern

Strg + Mausrad zurück

Verkleinern

Strg + 0 (Null)

Zoomfaktor zurücksetzen

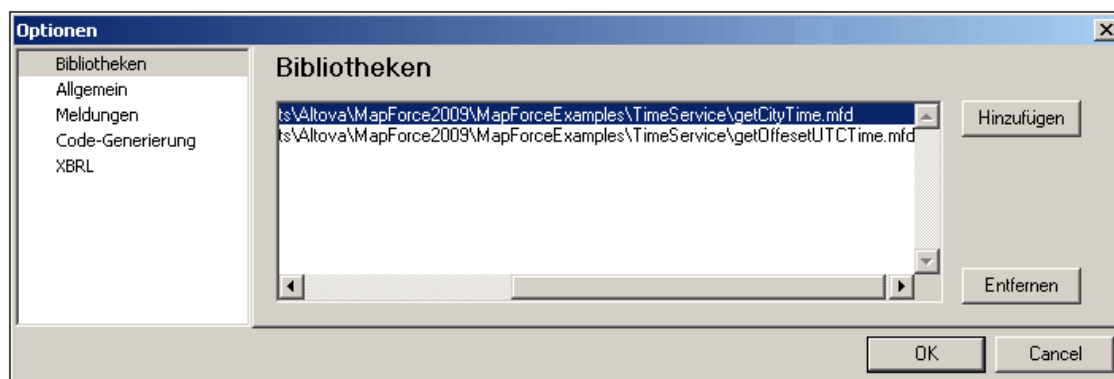
**Optionen**

Öffnet das Dialogfeld "Optionen" in dem Sie folgende Möglichkeiten haben:

- Hinzufügen oder Löschen von [benutzerdefinierten XSLT-Funktionen](#) oder [Bibliotheken](#).
- Definition **allgemeiner** Einstellungen wie z.B. Standard-Ausgabekodierung für neue Komponenten auf dem Register "Allgemein".
- Definieren, welche Meldungen reaktiviert werden sollen.
- Definition Ihrer spezifischen Kompilier- und IDE-Einstellungen.
- Definition der jeweiligen Anzeigeeinstellungen für XBRL-Komponenten

**Register "Bibliotheken":**

- Dient zum Hinzufügen oder Löschen von benutzerdefinierten XSLT- oder Programmiersprachenbibliotheken/-funktionen zu MapForce.

**Register "Allgemein":**

- Gibt an, ob das Logo, die Copyright-Informationen usw. beim Start und/oder Drucken angezeigt werden soll.
- Richtet Komponenten oder Funktionen beim Ziehen mit der Maus an anderen Komponenten aus.
- Aktivieren/Deaktivieren des Farbverlaufs im Hintergrund
- Beschränken des Annotationstexts in Komponenten auf X Zeilen. Gilt auch für



- SELECT-Anweisungen, die in einer Komponente zu sehen sind.
- Definition der Standardzeichencodierung für neue Komponenten
- Definition der Standardausgabekodierung für neue Komponenten
- Ausführungs-Timeout für das Register "Ausgabe" bei Anzeige der Vorschau des Mapping-Ergebnisses.
- Gibt an, ob die Ausgabe in **temporäre** Dateien erfolgen soll (Standardeinstellung) oder **direkt** auf die Festplatte geschrieben werden soll, wenn Sie auf die Schaltfläche/das Register "Ausgabe" klicken.

**Achtung:** Wenn Sie die Option "Direkt in die endgültigen Output-Dateien schreiben" aktivieren, werden Ausgabedateien überschrieben, ohne dass Sie dies vorher bestätigen müssen.

- Beschränkt die Ausgabe auf X Millionen Zeichen, wenn Sie den Code mit dem Built-In-Ausführungsprozessor generieren. Der Built-In-Ausführungsprozessor ist das einzige Ziel, das XML, CSV und FLF Streaming unterstützt.

**Allgemein**

Logo anzeigen  
☒ Beim Start  
☒ Beim Drucken

Mapping-Ansicht  
☐ Schattierten Hintergrund anzeigen  
☒ Anzeige der Annotation einschränken auf  Zeilen

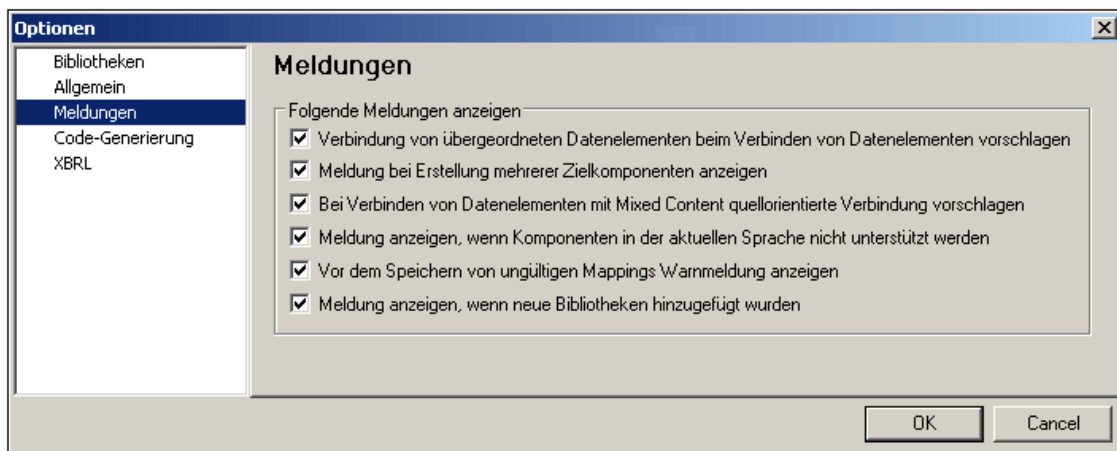
Standardkodierung für neue Komponenten  
Kodierungsname:   
Bytefolge:  ☐ Bytefolge-Markierung inkl.

Einstellungen Vorschau  
☐ Timeout benutzen:  Sek.  
Bei Aktivierung des Ausgaberegisters  
☒ Output-Datei als temporäre Datei generieren  
☐ Direkt in die endgültigen Output-Dateien schreiben  
Text in Schritten von  Millionen Zeichen anzeigen

OK Cancel

Register **Meldungen:**

Über dieses Register können Sie Meldungsfelder, die Sie zu einem früheren Zeitpunkt durch Aktivieren des Kontrollkästchens "Nicht mehr fragen" deaktiviert haben wieder aktivieren.

**Register "Generierung":**

Hier können Sie die Einstellungen für Ihre IDE und Ihren Compiler vornehmen.

**C++ Einstellungen:**

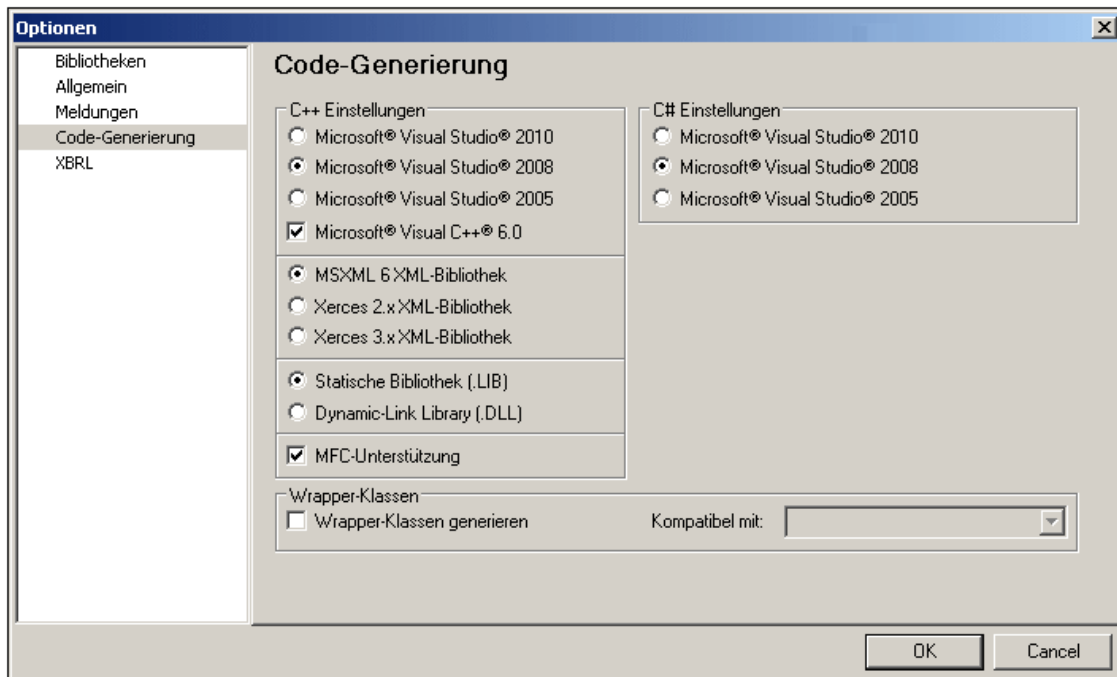
Definiert die spezifischen Kompiliereinstellungen für die C++-Umgebung.

**C# Einstellungen:**

Definiert die spezifischen Kompiliereinstellungen für die C#-Umgebung.

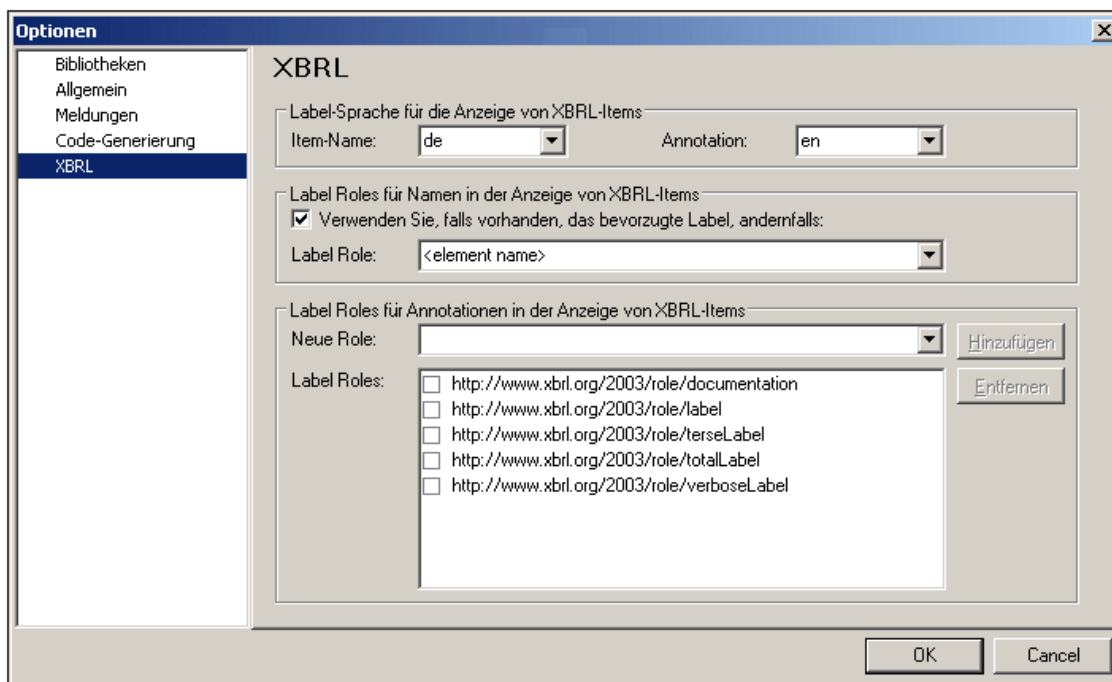
**Wrapper-Klassen**

Dient zum Auswählen der Versionskompatibilität der generierten Wrapper-Klassen für XML-Schemas. Derzeit verfügbare Optionen: V2005r3, 2006-2007, 2007r3 und höher. Diese Wrapper-Klassen können von benutzerdefiniertem Code verwendet werden, der den von MapForce generierten Code enthält.

**Register XBRL:**

Dient zum Definieren spezifischer Anzeigeeinstellungen von XBRL-Komponenten:

- Die Label-Sprache für die XBRL Items und ihre Annotationen.
- Die bevorzugten Label-Rollen für XBRL Item-Namen
- Den spezifischen Typ von Label-Rollen von Annotationen für XBRL Items



## 30.11 Fenster

### **Überlappend**

Mit diesem Befehl ordnen Sie alle offenen Dokumentfenster an, sodass sie übereinander gestapelt und überlappend angezeigt werden.

### **Horizontal anordnen**

Mit diesem Befehl ordnen Sie alle offenen Dokumente **horizontal nebeneinander** an, sodass alle Dokumentfenster gleichzeitig sichtbar sind.

### **Vertikal anordnen**

Mit diesem Befehl ordnen Sie alle offenen Dokumente **vertikal nebeneinander** an, sodass alle Dokumentfenster gleichzeitig sichtbar sind.

### **1**

### **2**

In dieser Liste werden alle offenen Fenster angezeigt, sodass Sie schnell zwischen ihnen wechseln können.

Sie können sich auch mit Strg-TAB oder Strg F6 durch die offenen Fenster schalten.

## 30.12 Hilfe

Im Menü **Hilfe** finden Sie Befehle zum Aufrufen der Bildschirmhilfe zu MapForce, Befehle zum Abrufen von Informationen über MapForce und Links zu den Support-Seiten auf der Altova Website. Außerdem enthält das Menü **Hilfe** das [Registrierungsdialogfeld](#), über das Sie nach Erwerb des Produkts den Lizenz-Keycode eingeben können.

Die Beschreibung der Befehle des Menüs **Hilfe** ist in die folgenden Unterabschnitte gegliedert:

- [Inhaltsverzeichnis, Index, Suchen](#)
- [Registrierung, Bestellformular](#)
- [Weitere Befehle](#)

### 30.12.1 Inhaltsverzeichnis, Index, Suchen

Mit dem Befehl **Inhaltsverzeichnis** wird die Bildschirmhilfe für MapForce geöffnet, wobei das Inhaltsverzeichnis auf der linken Seite des Hilfefensters angezeigt wird. Das Inhaltsverzeichnis bietet eine gute Übersicht über das gesamte Hilfedokument. Wenn Sie auf einen Eintrag im Inhaltsverzeichnis klicken, gelangen Sie zu diesem Thema.

Mit dem Befehl **Index** wird die Bildschirmhilfe für MapForce geöffnet, wobei auf der linken Seite des Hilfefensters der Stichwortindex angezeigt wird. Im Index werden Stichwörter aufgelistet und Sie können durch Doppelklick auf das Stichwort zum jeweiligen Thema navigieren. Wenn ein Stichwort mit mehr als einem Thema verknüpft ist, können Sie das Thema aus einer Liste von Themen auswählen.

Mit dem Befehl **Suchen** wird die Bildschirmhilfe für MapForce geöffnet, wobei auf der linken Seite des Hilfefensters das Suchdialogfeld angezeigt wird. Um nach einem Begriff zu suchen, geben Sie ihn in das Eingabefeld ein und drücken Sie die Eingabetaste. Das Hilfesystem führt eine Volltextsuche am gesamten Hilfedokument durch und gibt eine Trefferliste zurück. Doppelklicken Sie auf einen Eintrag, um das Thema anzuzeigen.

### 30.12.2 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 von MapForce 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.



### 30.12.3 Weitere Befehle

Der Befehl **Support Center** ist ein Link zum Altova Support Center im Internet. Im Support Center finden Sie Antworten auf häufig gestellte Fragen, Diskussionsforen, in denen Sie Software-Probleme besprechen können und ein Formular, um unsere Mitarbeiter vom technischen Support zu kontaktieren.

Der Befehl **Fragen und Antworten im Web** ist ein Link zur Altova-Fragen- und Antworten-Datenbank im Internet. Diese Datenbank wird ständig aktualisiert, wenn die Mitarbeiter von Altova von unseren Kunden mit neuen Fragen konfrontiert werden.

Der Befehl **Komponenten downloaden** ist ein Link zum Komponenten Download Center von Altova im Internet. Von hier können Sie Software-Komponenten verschiedener anderer Anbieter herunterladen, die Sie mit Altova Produkten verwenden können. Dabei handelt es sich um XSLT- und XSL-FO-Prozessoren, Applikationsserverplattformen usw. Die im Komponenten Download Center verfügbare Software ist normalerweise kostenlos.

Der Befehl MapForce **im Internet** ist ein Link zur [Altova Website](#) im Internet. Hier erfahren Sie mehr über MapForce und verwandte Technologien und Produkte auf der [Altova Website](#).

Der Befehl **MapForce Training** ist ein Link zur Online Training-Seite auf der [Altova Website](#). Hier können Sie aus Online-Kursen, die von ausgebildeten Altova Experten gehalten werden, auswählen.

Mit dem Befehl **Über** MapForce wird das Willkommensfenster und die Versionsnummer Ihres Produkts angezeigt



# Kapitel 31

---

## Code Generator

## 31 Code Generator

MapForce enthält einen integrierten Code Generator, der automatisch Java-, C++ oder C#-Klassendateien anhand von XML-Schemadefinitionen, Textdateien, Datenbanken, UN/EDIFACT-Dateien und ASC X12-Dateien generieren kann.

Dabei beschränkt MapForce das Mappen nicht auf 1:1-Beziehungen, sondern gestattet die gleichzeitige Verwendung mehrerer Quellen und Ziele, wobei jede beliebige Kombination unterschiedlicher Datenquellen in einer gemischten Umgebung gemappt werden kann.

Bei der Code Generierung wird ein voll funktionsfähiger und vollständiger Software-Code generiert, der das Mapping für Sie durchführt. Sie können den generierten Code in Ihre eigene Applikation einfügen oder ihn durch Ihre eigenen Funktionen ergänzen.

## 31.1 Einführung in Code Generator

Im Fall von XML-Schemas generiert die Standardvorlage von MapForce Code Generator automatisch Klassendefinitionen für alle deklarierten Elemente oder complexTypes. Diese Definitionen definieren alle complexTypes in Ihrem Schema neu, wobei die durch die Erweiterungen der complexTypes in Ihrem XML-Schema definierten Klassenableitungen sowie alle für das Mapping erforderlichen Klassen erhalten bleiben.

Bei komplexen Schemas, die Schemakomponenten aus mehreren Namespaces importieren, behält MapForce diese Informationen durch Generierung der entsprechenden C#-, C++-Namespaces oder Java-Pakete bei.

Zusätzlicher Code wird implementiert, wie z.B. Funktionen, die XML-Dateien in eine speicherresidente DOM (Document Object Model)-Repräsentation auslesen, XML-Dateien von einer DOM-Repräsentation zurück in eine Systemdatei schreiben, sowie Funktionen, die Strings in XML-DOM-Strukturen konvertieren und umgekehrt.

Der erzeugte Programmcode wird in den Programmiersprachen C++, Java oder C# ausgegeben.

| Zielsprache                         | C++                                                                                                             | C#                                                                                           | Java                                                           |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|----------------------------------------------------------------|
| <b>Entwicklungs-umgebungen</b>      | Microsoft Visual C++ 2010<br>Microsoft Visual C++ 2008<br>Microsoft Visual C++ 2005<br>Microsoft Visual C++ 6.0 | Microsoft Visual Studio 2010<br>Microsoft Visual Studio 2008<br>Microsoft Visual Studio 2005 | Apache Ant (build.xml file)<br>Eclipse 3.4<br>Borland JBuilder |
| <b>XML DOM Implementierungen</b>    | MSXML 6.0 oder höher<br>Apache Xerces 2.6 oder höher                                                            | System.Xml                                                                                   | JAXP                                                           |
| <b>Datenbank API (nur MapForce)</b> | ADO                                                                                                             | ADO.NET                                                                                      | JDBC                                                           |

### C++

Der generierte C++-Ausgabecode verwendet entweder MSXML 6.0 oder Apache Xerces 2.6 oder höher. Bei der C++-Codegenerierung werden sowohl in MapForce als auch in XMLSpy komplette Projekt- und Solution/Workspace-Dateien direkt für Visual C++ 6.0 oder Visual Studio 2005 bis 2010 generiert.

Zusätzlich zu den **.dsw** / **.dsp**-Dateien für Visual Studio 6.0 werden auch immer eine **.sln** und **.vcproj**-Datei generiert. Der generierte Code unterstützt optional MFC - z.B. durch Konvertierungsmethoden für generierten CString.

#### Bitte beachten Sie:

Beim Erstellen von C++-Code für Visual Studio 2005/2008/2010 und bei Verwendung einer für Visual C++ 6.0 vorkompilierten Xerces-Bibliothek muss eine Compiler-Einstellung in allen Projekten der Applikation geändert werden:

1. Wählen Sie alle Projekte im Explorer der Applikation aus.
2. [Project] | [Properties] | [Configuration Properties] | [C/C++] | [Language]
3. Wählen Sie *All Configurations*
4. Ändern Sie **Treat wchar\_t as Built-in Type** in **No (/Zc:wchar\_t)**

### C#

Der generierte C#-Code verwendet die .NET XML-Klassen (System.Xml) und kann von jeder .NET-fähigen Programmiersprache verwendet werden, z.B. VB.NET, Managed C++, J# oder eine der vielen Sprachen für die .NET-Plattform. Projektdateien können für Visual Studio 2005, 2008 und 2010 generiert werden.

**Java**

Der generierte Java-Code wird anhand des Industriestandards "Java API for XML Parsing" (JAXP) geschrieben und enthält eine JBuilder-Projektdatei, eine Ant build-Datei sowie Projektdateien für Eclipse. Unterstützt werden Java Versionen ab Java 5 (JDK 1.5).

Resultat in MapForce:

| <b>Resultat</b>                         | <b>Speicherort</b>                                                                  | <b>MapForce</b>                                                            |
|-----------------------------------------|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| Standardbibliotheken                    | Ordner "Altova"                                                                     | <input checked="" type="checkbox"/>                                        |
| Schema Wrapper-Bibliotheken             | Schema-Name-Ordner                                                                  | <input checked="" type="checkbox"/>                                        |
| Datenbank-Type-Info-Bibliotheken        | Datenbank-Name-Ordner                                                               | <input checked="" type="checkbox"/>                                        |
| EDI-Type-Info-Bibliotheken              | EDI-Message-Name-Ordner                                                             | <input checked="" type="checkbox"/>                                        |
| <b>Applikation</b>                      |                                                                                     |                                                                            |
| Mapping-Applikation<br>(komplette App.) | Kompilierung und Ausführung.<br>Führt das definierte Mapping<br>durch.              | <input checked="" type="checkbox"/>                                        |
| Mapping-Applikation kann nun:           | durch den Benutzer erweitert oder<br>in die eigene Applikation importiert<br>werden | <input checked="" type="checkbox"/><br><input checked="" type="checkbox"/> |

**Code Generator-Vorlagen**

Ausgabecode kann über eine einfache aber leistungsfähige Template-Sprache vollständig vom Benutzer angepasst werden. Diese Template-Sprache bietet volle Kontrolle über das Mappen von in XML-Schemas integrierten Datentypen auf die primitiven Datentypen einer bestimmten Programmiersprache.

Sie können damit den zugrunde liegenden Parse- und Validierungsprozessor problemlos ersetzen, Code entsprechend den Codekonventionen Ihres Unternehmens anpassen oder verschiedene Basisbibliotheken wie z.B. die Microsoft Foundation Classes (MFC) und die Standard Template Library (STL) verwenden.

## 31.2 Was ist neu ...

### Version 2011

Enthält Bug Fixes und Verbesserungen.

### Version 2010 R3

- Unterstützung für Microsoft Visual Studio 2010
- Unterstützung für MSXML 6.0 im generierten C++-Code
- Unterstützung für 64-Bit-Zielkomponenten für C++- und C#-Projekte

### Version 2010 R2

- 

### Version 2010

- Enumeration Facets aus XML-Schemas stehen nun als Symbolkonstanten in den generierten Klassen zur Verfügung (bei Verwendung von 2007r3 Vorlagen).

### Version 2009 sp1

- Wurde um [Apache Xerces Version 3.x](#) Unterstützung erweitert. (auch die älteren Versionen ab Xerces 2.6.x werden weiterhin unterstützt.)

### Version 2009

- Die generierte Mapping-Implementierung wurde überarbeitet und unterstützt nun Sequenzen und Gruppierungen. Die API hat sich nicht geändert.
- 

### Version 2008 R2

- Unterstützung für die Generierung von Visual Studio 2008 Projektdateien für C# und C++.
- Generierter MapForce Mapping Code in C# und Java kann Readers/Writers, Streams, Strings oder DOM-Dokumente als Quellen und Ziele verwenden.

### Version 2008

Die neuen 2007 SPL-Vorlagen im R3-Stil wurden weiter verbessert:

- Es können nun einzelne Elemente entfernt werden
- Sie haben Zugriff auf Schema-Metadaten (z.B. Elementnamen, Facets, Enumerationen, Occurrence usw.)
- ComplexType, die durch Erweiterungen abgeleitet werden, werden nun als abgeleitete Klassen generiert

### Version 2007 R3

Der Code Generator wurde in Version 2007 Release 3 zur Vereinfachung des generierten Codes, der Verringerung des Codevolumens und zur Verbesserung der Leistung überarbeitet.

- XML-Nodes sind jetzt eindeutig DOM-Dokumenten zugeordnet, um Probleme mit Speicherverwaltung und Multithreading zu vermeiden
- Neue Syntax zur Vermeidung von Kollisionen
- Neue Datentypen zur einfacheren Verwendung und Leistungsoptimierung (wenn möglich "Native Types", neue Behandlung von Null, ...)

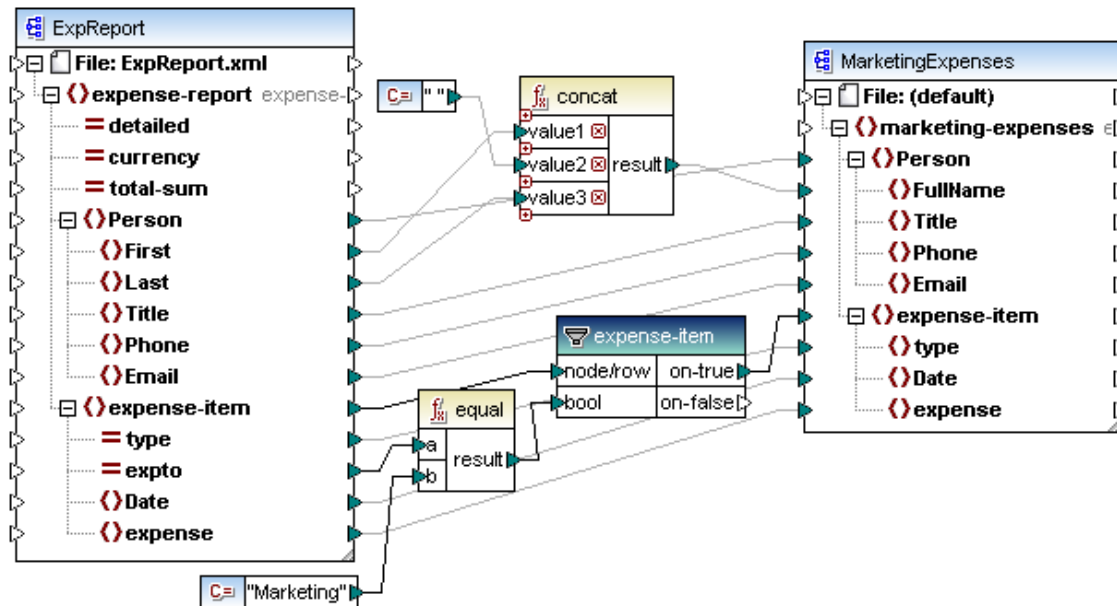
- Attribute werden nicht mehr als Sammlungen generiert
- Der Inhalt von Simple Elements wird nun aus Konsistenzgründen ebenfalls wie ein spezielles Attribut behandelt
- Neues internes Objektmodell (wichtig für benutzerdefinierte SPL-Vorlagen)
- Kompatibilitätsmodus zur Generierung von Code im Stile älterer Versionen
- Type Wrapper-Klassen werden nun nur mehr auf Anweisung für kleinere Codefragmente generiert



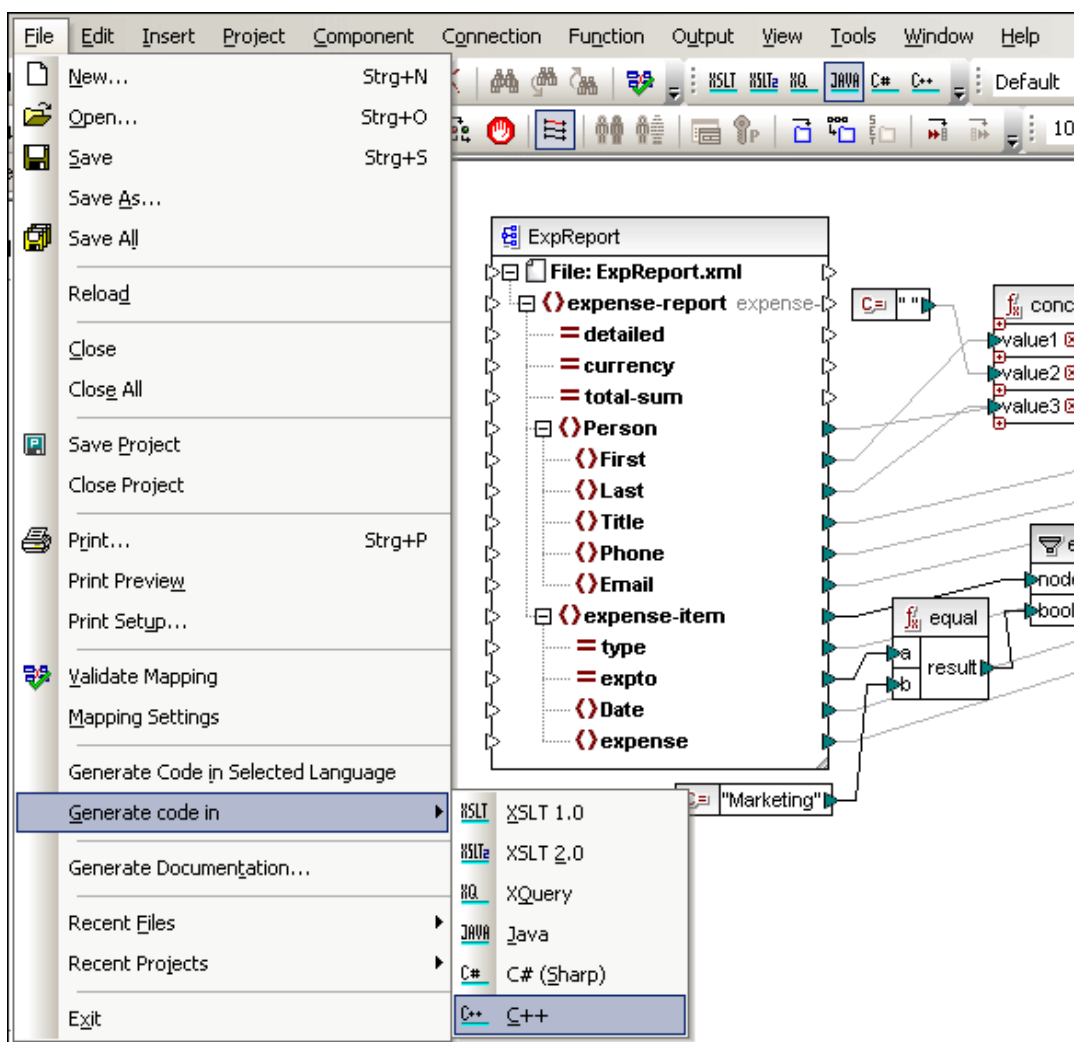
### 31.3 Erzeugen von Programmcode

In diesem Beispiel werden die allgemeinen Schritte zur Generierung von Programmcode über MapForce beschrieben. Dazu wird die Datei MarketingExpenses.mfd aus dem Ordner ...\\MapForceExamples verwendet. Nähere Informationen dazu finden Sie des Weiteren unter [Integrieren von MapForce Code in Ihre Applikation](#).

1. Öffnen Sie die Datei **MarketingExpenses.mfd**.



2. Wählen Sie die Menüoption **Datei | Code generieren in | C++**.



Daraufhin wird das Dialogfeld **Ordner suchen** geöffnet.

3. Navigieren Sie zu dem Ordner, in dem der Code gespeichert werden soll, und klicken Sie zur Bestätigung auf OK.  
Die Meldung "C++-Code-Generierung abgeschlossen" wird angezeigt.
4. Bestätigen Sie die Meldung mit OK.

Der generierte Code wird in Unterverzeichnissen unterhalb des angegebenen Verzeichnisses platziert. Er enthält alle Bibliotheken usw., die für die Kompilierung und Ausführung des Mapping-Codes benötigt werden.

Die hier beschriebenen Schritte werden an einer späteren Stelle in diesem Dokument erneut beschrieben und durch zusätzliche Informationen zur Erzeugung und Kompilierung der einzelnen Programmiersprachen ergänzt:

Nähere Informationen finden Sie unter den folgenden Abschnitten:

[Generieren von Java-Code](#)

[Generieren von C#-Code](#)

[Generieren von C++-Code](#)

### 31.3.1 Generieren von Java-Code

Voraussetzungen und Standardeinstellungen:

Die generierte MapForce-Applikation unterstützt Java 5 oder höher.

Damit Java-Code generiert werden kann, wenn Datenbankdaten gemappt wurden, müssen **JDBC-Treiber** installiert sein. Nähere Informationen dazu finden Sie im Abschnitt [JDBC-Verbindungen](#).

Über die Menüoption **Datei | Mapping-Einstellungen** können die Einstellungen für das Mapping-Projekt definiert werden. Die Standardeinstellungen sind:

- Applikationsname=Mapping
- Base Package Name=com.mapforce

Um die JDBC-spezifischen Einstellungen zu sehen, klicken Sie auf die Datenbankkomponente und wählen Sie die Menüoption **Komponente | Eigenschaften**.

In der nachstehenden Tabelle werden die verschiedenen Applikationen angezeigt, die in den einzelnen Umgebungen kompiliert werden können:

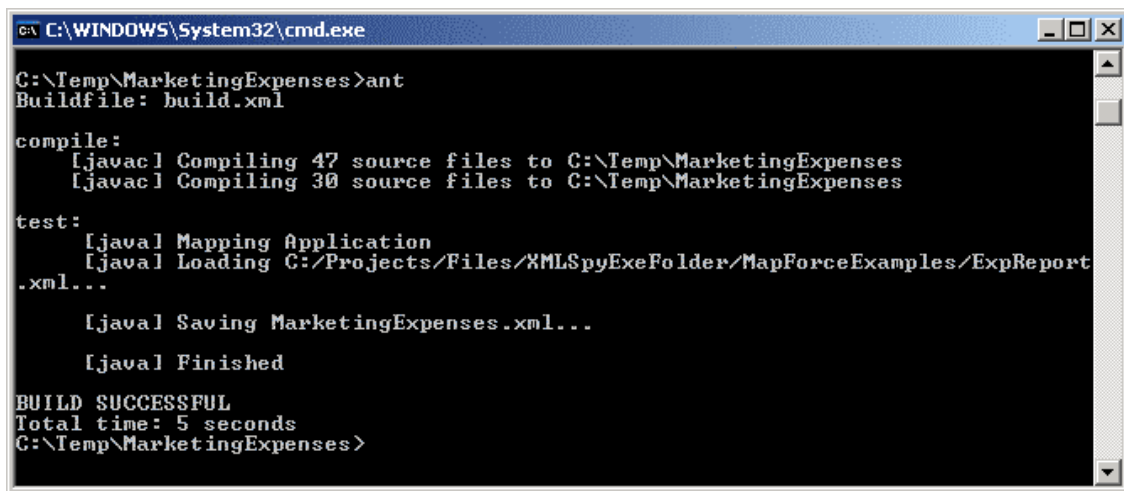
| Dateiname (mit den Namen der Standardapplikation) | Anmerkung           |
|---------------------------------------------------|---------------------|
| MappingConsole.java                               | Konsolenapplikation |
| MappingApplication.java                           | Dialog-Applikation  |

#### So generieren Sie Java-Code in MapForce:

1. Wählen Sie die Menüoption **Datei | Code generieren in | Java**.  
Sie werden daraufhin aufgefordert, das Zielverzeichnis für die generierten Dateien anzugeben.
2. Wählen Sie das gewünschte Verzeichnis aus und klicken Sie zur Bestätigung auf OK (z.B. c:\codegen\Java).  
Wenn der Vorgang erfolgreich durchgeführt wurde, wird die Meldung "Java Codegenerierung wurde erfolgreich ausgeführt" angezeigt. Andernfalls wird eine entsprechende Fehlermeldung angezeigt.

Bei Verwendung eines **Ant** Build Script:

- Navigieren Sie zum Java-Unterverzeichnis und führen Sie "ant" aus (Daraufhin wird automatisch die Datei **build.xml** geöffnet)
- Daraufhin wird der Java-Code kompiliert und ausgeführt. Am Ende dieser Sequenz wird automatisch das XML-Instanzziel generiert.



```
C:\WINDOWS\System32\cmd.exe

C:\Temp\MarketingExpenses>ant
Buildfile: build.xml

compile:
[javac] Compiling 47 source files to C:\Temp\MarketingExpenses
[javac] Compiling 30 source files to C:\Temp\MarketingExpenses

test:
[java] Mapping Application
[java] Loading C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml...
[java] Saving MarketingExpenses.xml...
[java] Finished

BUILD SUCCESSFUL
Total time: 5 seconds
C:\Temp\MarketingExpenses>
```

Nähere Informationen dazu finden Sie unter:

[Generieren von Java-Code mittels JBuilder](#)

### 31.3.2 Generieren von C#-Code

Voraussetzungen und Standardeinstellungen:

Die Menüoption **Datei | Mapping-Einstellungen** definiert die Einstellungen für das Mapping-Projekt. Die Standardeinstellungen sind:

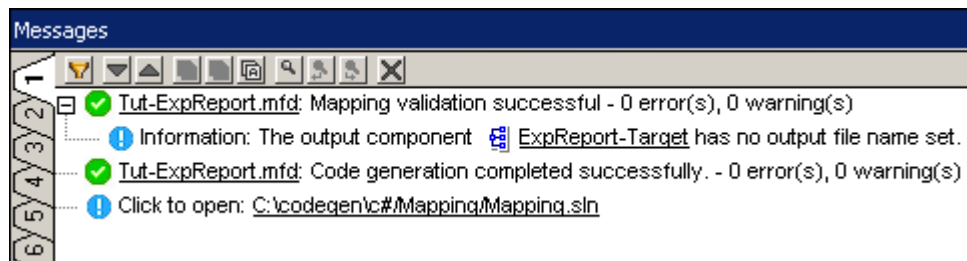
- Applikationsname=Mapping

Um Datenbank-spezifische Einstellungen anzuzeigen, klicken Sie auf eine Datenbank-Komponente und wählen Sie die Menüoption **Komponente | Einstellungen**.

**.sln** und **csproj** Dateien werden für Microsoft Visual Studio generiert. Über das Register **Generierung** der Menüoption **Extras | Optionen** können Sie die Version der Ziel-IDE auswählen. Benutzer von Mono können Visual Studio Solution-Dateien mit xbuild von Mono verwenden.

#### So generieren Sie C#-Code in MapForce:

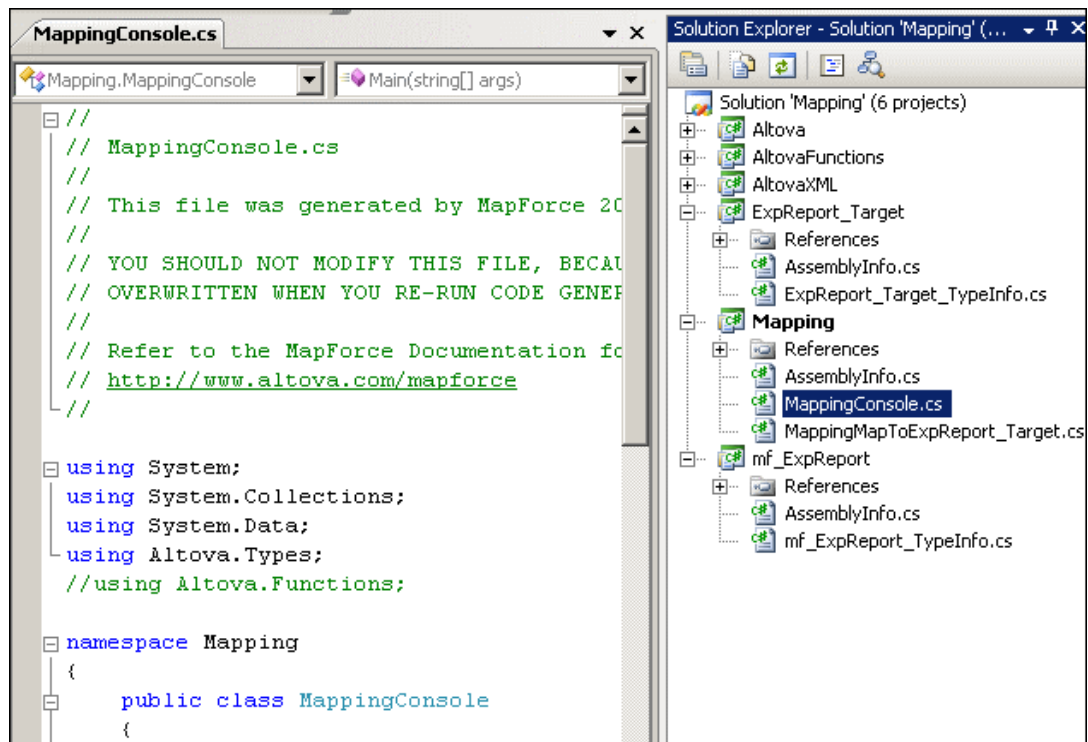
1. Wählen Sie die Menüoption **Datei | Code generieren in | C# (sharp)**  
Sie werden daraufhin aufgefordert, das Zielverzeichnis für die erzeugten Dateien anzugeben.
2. Wählen Sie das Verzeichnis, in dem die Dateien gespeichert werden sollen (z.B. `c:\codegen\C#`) und bestätigen Sie mit OK.  
Die Meldung "C#-Code-Generierung abgeschlossen" wird angezeigt, wenn der Vorgang erfolgreich war. Andernfalls erscheint eine Fehlermeldung mit einer Beschreibung des/der jeweiligen Fehler(s).



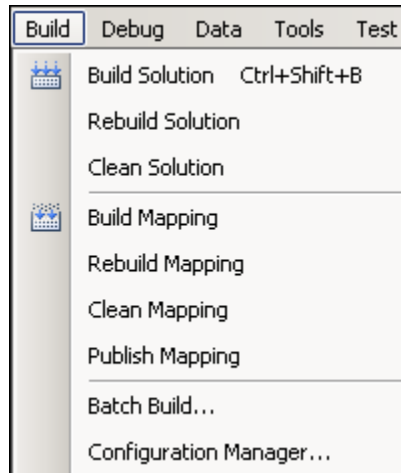
#### Ordner `c:\codegen\C#\mapping`

Sie können das Projekt in Microsoft Visual Studio kompilieren.

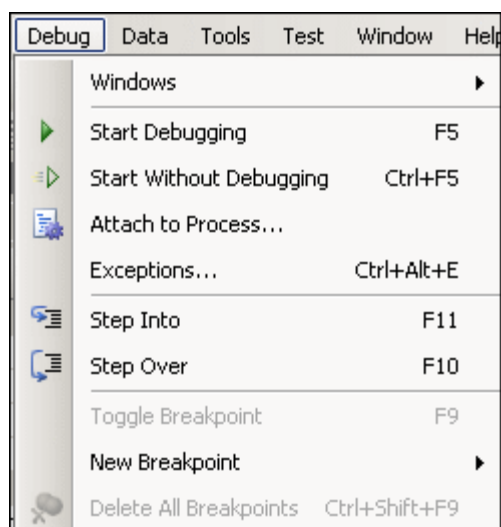
- Mapping.sln (für Visual Studio)
1. Klicken Sie im Fenster "Meldungen" von Mapforce auf den Link "**Click to open: C:\codegen\C#\Mapping\Mapping.sln**" um die Solution-Datei zu öffnen. Der Link "Click to open..." steht für C# und C++ zur Verfügung. Wenn Sie für Java kompilieren, müssen Sie manuell zum entsprechenden Ordner navigieren.



- Wählen Sie die Menüoption **Build | Build Solution** um das Mapping-Projekt zu kompilieren.



- Wählen Sie die Menüoption **Debug | Start Debugging** um die Applikation zu starten.



Die Mapping-Applikation wird gestartet und die XML-Zielfdatei wird erstellt.

### 31.3.3 Generieren von C++-Code

Voraussetzungen und Standardeinstellungen:

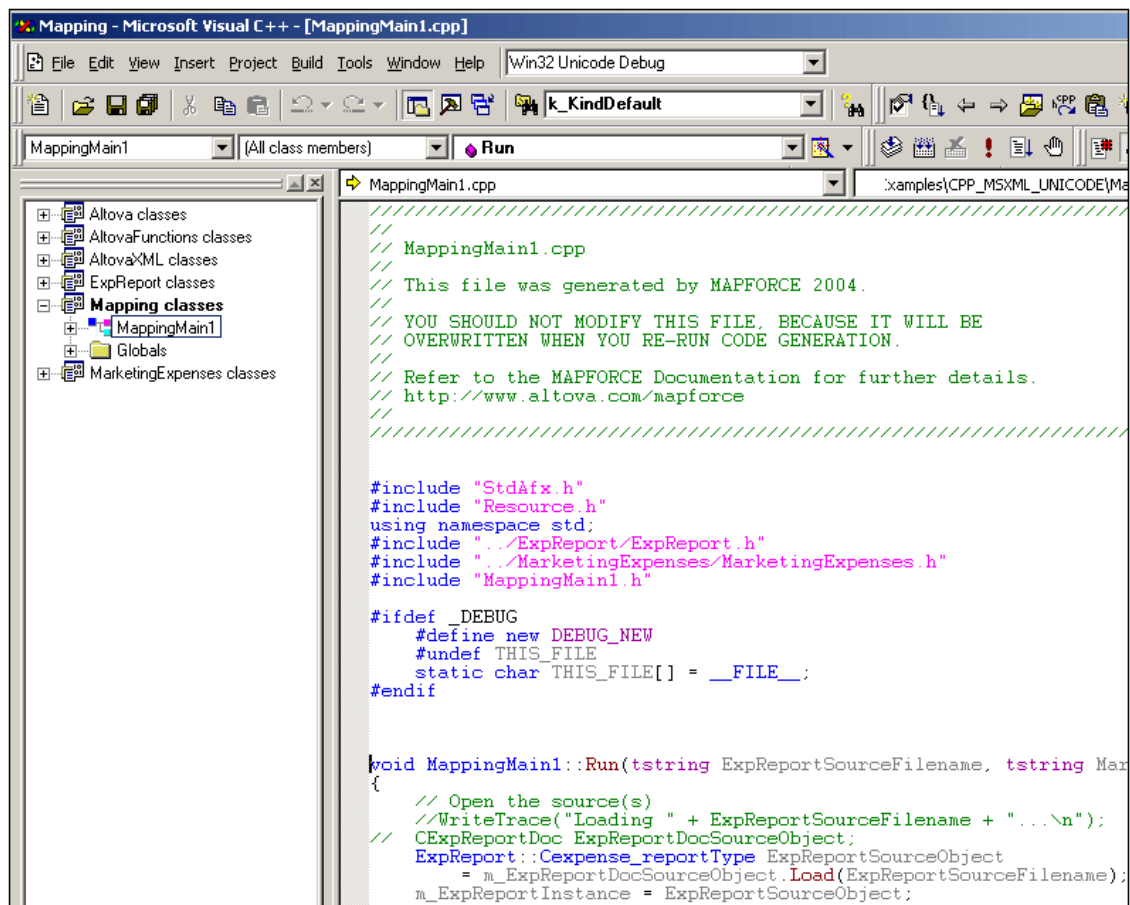
- Bei der C++-Codegenerierung werden sowohl in MapForce als auch in XMLSpy Visual C++ 6.0, 7.1 / Visual Studio 2005, 2008 und 2010 direkt unterstützt.
- Zusätzlich zu den **.dsw** / **.dsp**-Dateien für Visual Studio 6.0 werden auch immer eine **.sln** und **.vcproj**-Datei generiert. Auf dem Register **Generierung** unter der Menüoption **Extras | Optionen** können Sie die Einstellungen für die IDE und die Codegenerierung festlegen.
- Die Menüoption **Datei | Mapping-Einstellungen** definiert die Einstellungen für das Mapping-Projekt. Die Standardeinstellungen sind Applikationsname=Mapping.

Um Datenbank-spezifische Einstellungen anzuzeigen, klicken Sie auf eine Datenbank-Komponente und wählen Sie die Menüoption **Komponente | Einstellungen**.

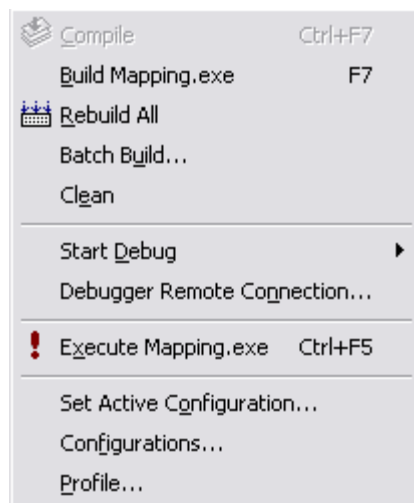
#### So generieren Sie C++-Code in MapForce:

1. Wählen Sie die Menüoption **Datei | Code generieren in | C++**.  
Sie werden daraufhin aufgefordert, das Zielverzeichnis für die erzeugten Dateien anzugeben.
2. Wählen Sie das Verzeichnis, in dem die Dateien gespeichert werden sollen (z.B. c:\codegen\C++) und bestätigen Sie mit OK.  
Die Meldung "C++-Code-Generierung abgeschlossen" wird angezeigt, wenn der Vorgang erfolgreich war. Andernfalls erscheint eine Fehlermeldung mit einer Beschreibung des/der jeweiligen Fehler(s).
3. Öffnen Sie die Datei **mapping.dsw** oder **mapping.sln** im Unterordner **Mapping**, also c:\codegen\C++\Mapping in Microsoft Visual C++.





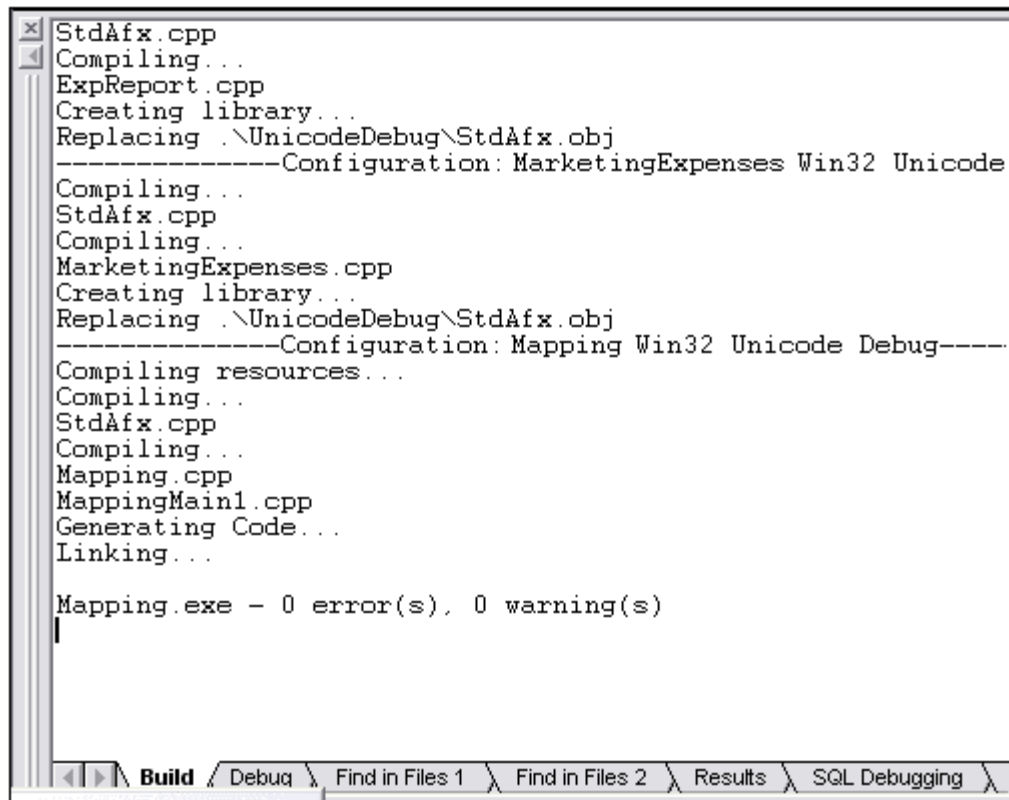
4. Wählen Sie die Menüoption **Build | Build Mapping.exe.**



Sie haben die Wahl zwischen vier verschiedenen Build-Konfigurationen. Beachten Sie bitte, dass nur Unicode Builds den vollständigen Unicode-Zeichensatz in XML- und anderen Dateien unterstützen. Nicht-Unicode-Builds arbeiten mit der lokalen Codepage Ihrer Windows-Installation.

|        |                  |
|--------|------------------|
| Debug: | Debug Unicode    |
|        | Debug NonUnicode |

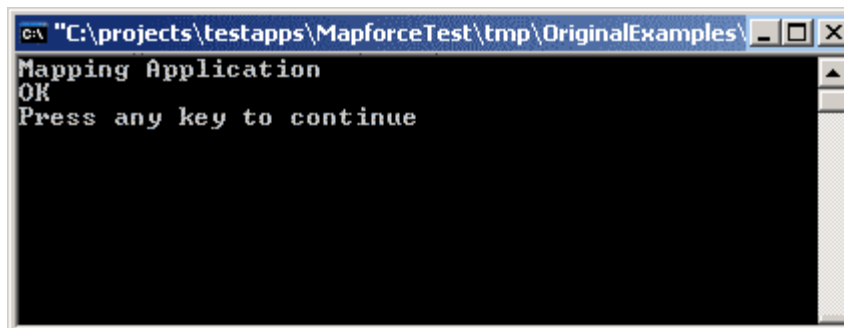
Release: Release Unicode  
Release Non Unicode



```
StdAfx.cpp
Compiling...
ExpReport.cpp
Creating library...
Replacing .\UnicodeDebug\StdAfx.obj
-----Configuration: MarketingExpenses Win32 Unicode
Compiling...
StdAfx.cpp
Compiling...
MarketingExpenses.cpp
Creating library...
Replacing .\UnicodeDebug\StdAfx.obj
-----Configuration: Mapping Win32 Unicode Debug-----
Compiling resources...
Compiling...
StdAfx.cpp
Compiling...
Mapping.cpp
MappingMain1.cpp
Generating Code...
Linking...

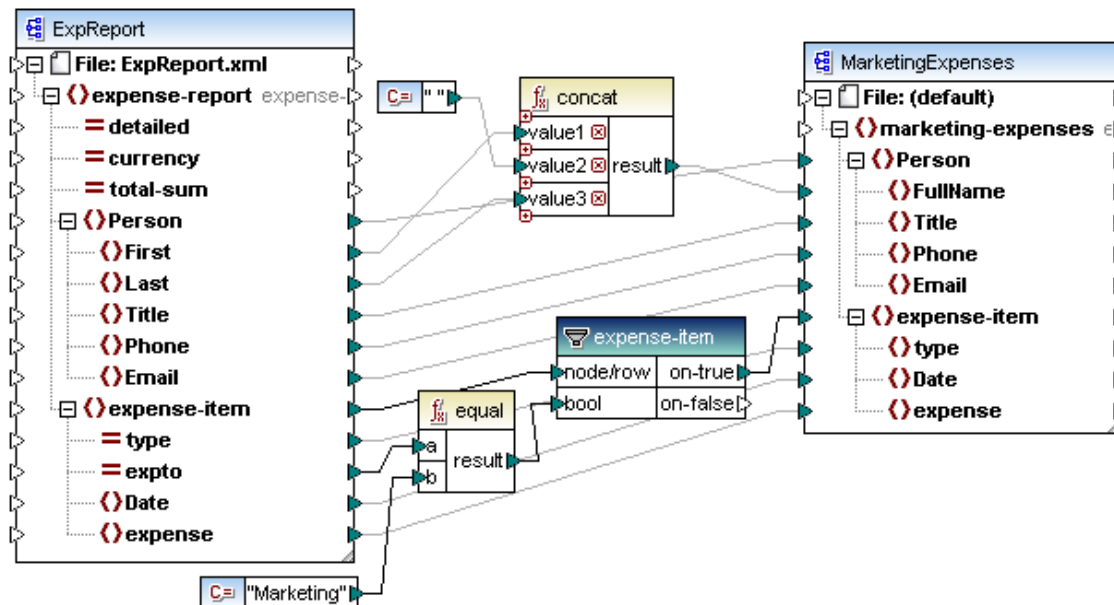
Mapping.exe - 0 error(s), 0 warning(s)
```

5. Nachdem der Code erzeugt worden ist, führen Sie die Datei **Mapping.exe** aus, um Ihre Daten zu mappen.



## 31.4 Mapping-Beispiel mit Generierung von Code

Im unten gezeigten Mapping-Beispiel arbeiten wir mit dem Mapping-Projekt MarketingExpenses.mfd im Ordner ...\\MapForceExamples. Weitere Informationen finden Sie auch unter [Integrieren von MapForce Code in Ihre Applikation](#).



- Der unten stehende Code definiert den Pfad der **Quelldatei** sowie die beim Mapping erzeugte **XML-Zieldatei**. Es können mehrere **Quelldateien** vorkommen, jedoch kann nur eine Zieldatei damit verknüpft werden.

```
MappingMapToMarketingExpensesObject.run(
```

```
"C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
"MarketingExpenses.xml");
```

- Existieren **mehrere Zieldateien**, so wird der Abschnitt MappingMain1 für jede Zieldatei **wiederholt**.

```
MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
MappingMapToMarketingExpensesObject.registerTraceTarget(ttc);
MappingMapToMarketingExpensesObject.run(
"C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
"MarketingExpenses.xml"
);
```

- Im Abschnitt Exception können Sie zusätzlichen Code zu Behandlung von Fehlern einfügen:  
`catch (Exception e), or`  
`catch (AltovaException e)`

Bei den unten stehenden Code-Fragmenten handelt es sich um den für die einzelnen Programmiersprachen jeweils generierten Mapping-Code. Sie können ganz einfach angepasst (z.B. um Dateinamen über die Befehlszeile zu akzeptieren) oder in andere Programme integriert werden.

**Java (com/mapforce/MappingConsole.java)**

```

public static void main(String[] args) {
 System.out.println("Mapping Application");
 try { // Mapping
 TraceTargetConsole ttc = new TraceTargetConsole();
 MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
 MappingMapToMarketingExpensesObject.registerTraceTarget(ttc);
 MappingMapToMarketingExpensesObject.run(
 "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
 "MarketingExpenses.xml"
);
 System.out.println("Finished");
 }
 catch (com.altova.UserException ue) {
 System.out.print("USER EXCEPTION:");
 System.out.println(ue.getMessage());
 System.exit(1);
 }
 catch (Exception e) {
 System.out.print("ERROR:");
 System.out.println(e.getMessage());
 e.printStackTrace();
 System.exit(1);
 }
}

```

#### C# (Mapping/MappingConsole.cs)

```

public static void Main(string[] args)
{
 try
 {
 TraceTargetConsole ttc = new TraceTargetConsole();
 MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
 MappingMapToMarketingExpensesObject.RegisterTraceTarget(ttc);
 MappingMapToMarketingExpensesObject.Run(
 "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
 "MarketingExpenses.xml");
 Console.Out.WriteLine("Finished");
 }
 catch (Altova.UserException ue)
 {
 Console.Out.Write("USER EXCEPTION: ");
 Console.Out.WriteLine(ue.Message);
 System.Environment.Exit(1);
 }
 catch (Exception e)
 {
 Console.Out.Write("ERROR: ");
 Console.Out.WriteLine(e.Message);
 Console.Out.WriteLine(e.StackTrace);
 System.Environment.Exit(1);
 }
}

```

#### C++ (Mapping/Mapping.cpp)

```

int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
 tcout << _T("Mapping Application") << endl;

 try
 {
 CoInitialize(NULL);
 }
}

```

```
 {
 MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject;
 MappingMapToMarketingExpensesObject.Run(
 _T(
"C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml"),
 _T("MarketingExpenses.xml"));
 }
 CoUninitialize();

 tcout << _T("OK") << endl;
 return 0;
}
catch (CAltovaException& e)
{
 if (e.IsUserException())
 tcerr << _T("User Exception: ");
 else
 tcerr << _T("Error: ");
 tcerr << e.GetInfo().c_str() << endl;
 return 1;
}
catch (_com_error& e)
{
 tcerr << _T("COM-Error from ") << (TCHAR*)e.Source() << _T(":") <<
endl;
 tcerr << (TCHAR*)e.Description() << endl;
 return 1;
}
catch (std::exception& e)
{
 cerr << "Exception: " << e.what() << endl;
 return 1;
}
catch (...)
{
 tcerr << _T("Unknown error") << endl;
 return 1;
}
}
```

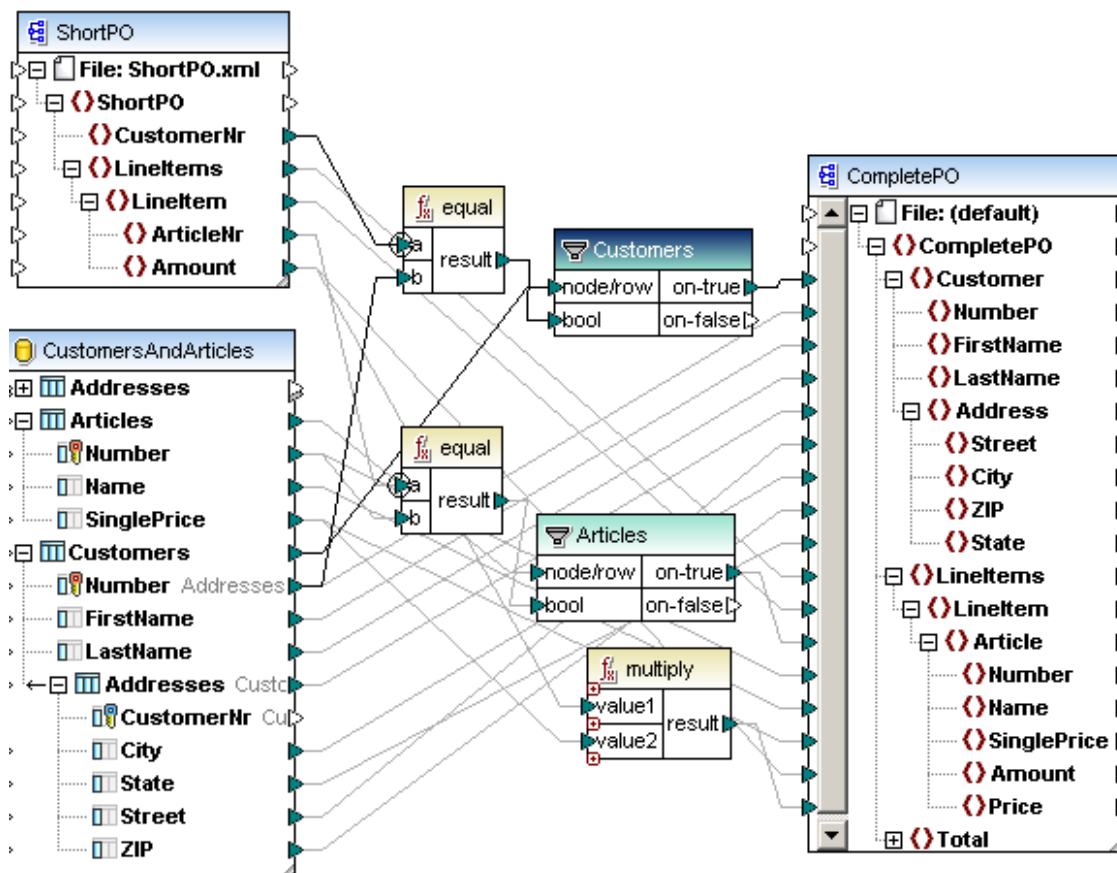
## 31.5 Integrieren von MapForce Code in Ihre Applikation

In MapForce generierter Code kann in Ihre jeweilige Applikation integriert oder dafür adaptiert werden. Bei der Code-Generierung wird eine komplette und voll funktionsfähige Applikation generiert. Das unten gezeigte Mapping-Projekt **DB\_CompletePO.mfd** steht im Ordner **...\\MapForceExamples** zur Verfügung.

Informationen dazu, wie der generierte Code erzeugt wird, finden Sie im Abschnitt [Erzeugen von Programmcode](#).

In diesem Abschnitt wird beschrieben wie Sie:

- die Quell- und Zieldatei eines Mapping-Projekts **ändern**
- einen **XML Input Stream** als Datenquelle verwenden und
- wo Sie Ihren eigenen **Fehlerbehandlungscode** hinzufügen



In diesem Beispiel werden zwei Quelldateien und eine Zieldatei verwendet:

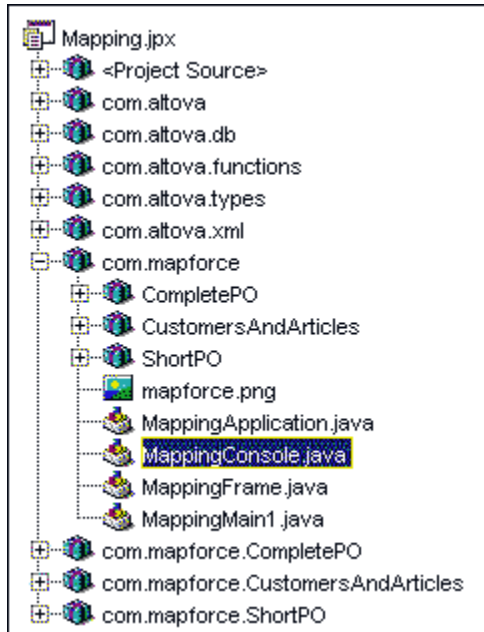
- ShortPO.xml als **XML-Quelldatei**
- CustomersAndArticles.mdb als **Quelldatenbank** und
- CompletePO.xml als **XML-Zieldatei**.

### 31.5.1 MapForce Code in Java-Applikationen

In diesem Beispiel wird davon ausgegangen, dass Sie **Borland JBuilder** als Ihre Java-Umgebung verwenden.

Nach Generierung des Java-Codes in MapForce und Definition eines Ordners "Output" als Ausgabeordner

1. Navigieren Sie zum Ordner **output\Mapping** und öffnen Sie die Projektdatei **Mapping.jpx**.
2. Doppelklicken Sie auf die Datei **MappingConsole.java**.



In der unten gezeigten Abbildung sehen Sie ein Codefragment.

```
{
 com.altova.io.Input ShortP02Source = new com.altova.io.FileInput(
"ShortPO.xml");
 com.altova.io.Output CompleteP02Target = new
com.altova.io.FileOutput("CompletePO.xml");

 MappingMapToCompleteP00bject.run(
 java.sql.DriverManager.getConnection(
 "jdbc:odbc;;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
 ""',
 ShortP02Source,
 CompleteP02Target);
}
```

Beachten Sie bitte, dass die Pfadnamen im erzeugten Quellcode aus Gründen der Übersichtlichkeit gelöscht wurden.

Wenn wir uns die Datei **MappingMapToCompletePO.run** ansehen, sehen wir, dass alle Parameter, die an die Methode **run** übergeben wurden, **mit Ausnahme** der letzten (CompletePO.xml) Quelldateien sind. Stellen Sie bitte sicher, dass der letzte Parameter die Zieldatei ist.

In diesem Beispiel:

sind die **Quelldateien**:

- XML-Datei: Short.PO.xml
- Datenbank: CustomerAndArticles.mdb einschließlich des Connection Strings.  
Die **beiden leeren Parameter ""** im Anschluss an die ersten Datenbankparameter dienen zur Eingabe von **Benutzernamen** und **Passwort** (unverschlüsselt) für jene Datenbanken, für die diese Daten erforderlich sind.

Die **Zieldatei** ist:

- CompletePO.xml

**So definieren Sie Ihre eigenen Quell- oder Zieldateien:**

- Bearbeiten Sie die Parameter, die an die FileInput- und FileOutput-Konstrukturen übergeben werden, direkt.

**So verwenden Sie einen XML Input Stream als XML-Datenquelle:**

- Ersetzen Sie FileInput durch StreamInput und übergeben Sie `java.io.InputStream` anstelle eines Dateinamens.

**So fügen Sie zusätzlichen Fehlerbehandlungscode hinzu:**

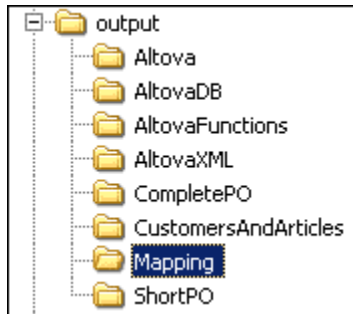
- Editieren Sie den Code unterhalb des `catch (Exception e)` Codes.



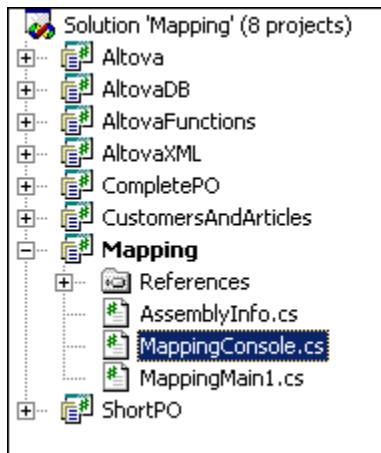
### 31.5.2 MapForce Code in C#-Applikationen

Nach Generierung des C#-Codes in MapForce und Definition eines Ordners "Output" als Ausgabeordner

1. Navigieren Sie zum Ordner output\**Mapping** und öffnen Sie die Projektdatei **Mapping.sln**.



2. Doppelklicken Sie auf die Datei **MappingConsole.cs**.



In der unten gezeigten Abbildung sehen Sie ein Codefragment.

```
{
 Altova.IO.Input ShortP02Source = new Altova.IO.FileInput(
"ShortP0.xml");
 Altova.IO.Output CompleteP02Target = new Altova.IO.FileOutput(
"CompleteP0.xml");

 MappingMapToCompleteP0Object.Run(
 "Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=CustomersAndArticles.mdb; ",
 ShortP02Source,
 CompleteP02Target);
}
```

Beachten Sie bitte, dass die Pfadnamen im erzeugten Quellcode aus Gründen der Übersichtlichkeit gelöscht wurden.

Wenn wir uns die Datei **MappingMapToCompletePO.Run** ansehen, sehen wir, dass alle Parameter, die an die Methode **run** übergeben wurden, **mit Ausnahme** der letzten (CompletePO.xml) Quelldateien sind. Stellen Sie bitte sicher, dass der letzte Parameter die Zieldatei ist.

In diesem Beispiel:

sind die **Quelldateien**:

- XML-Datei: Short.PO.xml
- Datenbank: CustomerAndArticles.mdb einschließlich des Connection Strings.

Die **Zieldatei** ist:

- CompletePO.xml

**So definieren Sie Ihre eigenen Quell- oder Zieldateien:**

- Bearbeiten Sie die Parameter, die an die FileInput- und FileOutput-Konstrukturen übergeben werden, direkt.

**So verwenden Sie einen XML Input Stream als XML-Datenquelle:**

- Ersetzen Sie FileInput durch StreamInput und übergeben Sie `System.IO.Stream` anstelle eines Dateinamens.

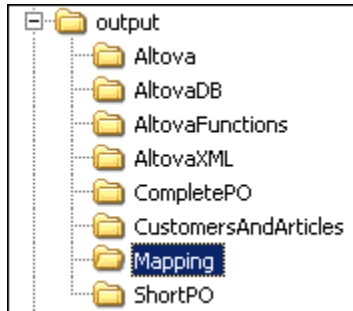
**So fügen Sie zusätzlichen Fehlerbehandlungscode hinzu:**

- Editieren Sie den Code unterhalb des `catch (Exception e)` Codes.

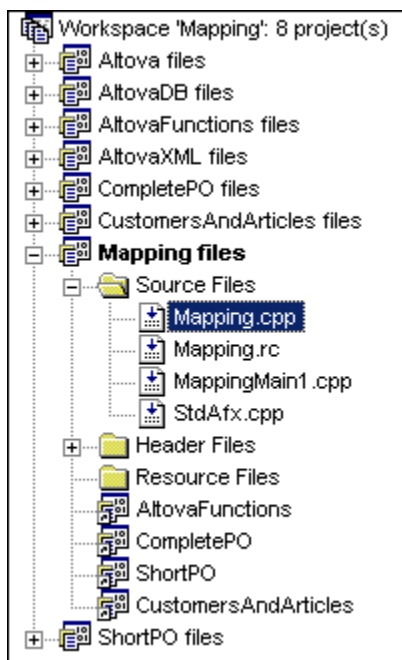
### 31.5.3 MapForce Code in C++-Applikationen

Nach Generierung des C++-Codes in MapForce und Definition eines Ordners "Output" als Ausgabeordner

1. Navigieren Sie zum Ordner **output\Mapping** und öffnen Sie die Workspace-Datei **Mapping.dsw** in MS Visual Studio 6.0 (oder **mapping.sln** in späteren Versionen von Visual C++)



2. Doppelklicken Sie auf die Datei **Mapping.cpp** um die Mapping-Projektdatei zu öffnen.



In der unten gezeigten Abbildung sehen Sie ein Codefragment.

```
CoInitialize(NULL);
{
 MappingMapToCompletePO MappingMapToCompletePOObject;
 MappingMapToCompletePOObject.Run(
 _T("Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=CustomersAndArticles.mdb; "),
 _T("ShortPO.xml"),
 _T("CompletePO.xml"));
}
CoUninitialize();

tcout << _T("Finshed") << endl;
return 0;
```

Beachten Sie bitte, dass die Pfadnamen im erzeugten Quellcode aus Gründen der Übersichtlichkeit gelöscht wurden.

Wenn wir uns die Datei **MappingMapToCompletePO.Run** ansehen, sehen wir, dass alle Parameter, die an die Methode **run** übergeben wurden, **mit Ausnahme** der letzten (CompletePO.xml) Quelldateien sind. Stellen Sie bitte sicher, dass der letzte Parameter die Zieldatei ist.

In diesem Beispiel:

sind die **Quelldateien**:

- XML-Datei: Short.PO.xml
- Datenbank: CustomerAndArticles.mdb einschließlich des Connection Strings.

Die **Zieldatei** ist:

- CompletePO.xml

**So definieren Sie Ihre eigenen Quell- oder Zieldateien:**

- Bearbeiten Sie die Parameter, die an die Run-Methode von MappingMapToCompletePOObject übergeben werden, direkt.

**So verwenden Sie einen XML Input Stream als XML-Datenquelle:**

- Navigieren Sie im Code zur Methodendeklaration von "run" und konfigurieren Sie dort die spezifischen Parameter.

**So fügen Sie zusätzlichen Fehlerbehandlungscode hinzu:**

- Editieren Sie den Code unterhalb des `catch (AltovaException& e) Codes`.

### 31.5.4 Unterstützung von Datenstreams

In C# und Java generierter Code unterstützt ab Version 2008 Release 2 zusätzlich zu dateibasiertem Input/Output auch Datenstreams als Input/Output.

Die wichtigste Funktion von generiertem Code ist die "run"-Funktion. Sie hat einen Parameter für jede statische Quelle oder Input-Komponente im Mapping und einen Endparameter für die Output-Komponente. Komponenten, die mehrere Dateien verarbeiten, werden nicht als Parameter für die "run"-Methode angezeigt, da die Dateinamen in diesem Fall dynamisch innerhalb des Mappings verarbeitet werden.

Vor MapForce Version 2008 R2 mussten bei der run-Funktion Dateinamen und/oder Datenbankverbindungsobjekte als Argumente verwendet werden. Nun können auch zusätzliche Input- oder Output-Objekte definiert werden.

Die folgenden Quellen und Ziele werden in jeder Kombination unterstützt:

- Dateien (identifiziert durch Dateinamen)
- Binär-Streams
- Bei Zeichen-Streams (readers/writers) - wird der Zeichenkodierungssatz, der in der MapForce-Komponente definiert ist, ignoriert
- Bei Strings wird der Zeichenkodierungssatz, der in der MapForce-Komponente definiert ist, ignoriert
- DOM-Dokumente (nur für XML)

Diese Objekte befinden sich im **com.altova.io** Paket (Java) und in den **Altova.IO** Namespaces (C#) und werden immer generiert. Die Namespace/Paket-Präfixe wurden aus Gründen der Übersichtlichkeit in den folgenden Abschnitten weggelassen.

Input- und Output-Objekte haben Basisklassen namens **Input** und **Output** also **com.altova.io.Input** und **com.altova.io.Output** für Java und **Altova.IO.Input** und **Altova.IO.Output** für C#.

Die unten verwendete Bezeichnung "**text**" steht für jede EDI- / X12- / HL7-, Flextext-, CSV- oder FLF-Datei.

Die folgenden Wrapper-Objekte werden als Parameter der "run" Methode verwendet:

#### Java:

Dateien/Dateinamen (XML, Text und Excel):

```
com.altova.io.FileInput(String filename)
com.altova.io.FileOutput(String filename)
```

Streams (XML, Text und Excel):

```
com.altova.io.StreamInput(java.io.InputStream stream)
com.altova.io.StreamOutput(java.io.OutputStream stream)
```

Strings (XML und Text):

```
com.altova.io.StringInput(String xmlcontent)
com.altova.io.StringOutput()
```

Java IO Reader/Writer (XML und Text):

```
com.altova.io.ReaderInput(java.io.Reader reader)
com.altova.io.WriterOutput(java.io.Writer writer)
```

DOM-Dokumente (nur XML):

```
com.altova.io.DocumentInput(org.w3c.dom.Document document)
```

```
com.altova.io.DocumentOutput(org.w3c.dom.Document document)
```

**C#:**

Dateien/Dateinamen (XML, Text und Excel):

```
Altova.IO.FileInput(string filename)
Altova.IO.FileOutput(string filename)
```

Streams (XML, Text und Excel):

```
Altova.IO.StreamInput(System.IO.Stream stream)
Altova.IO.StreamOutput(System.IO.Stream stream)
```

Strings (XML und Text):

```
Altova.IO.StringInput(string content)
Altova.IO.StringOutput(StringBuilder content)
```

Java IO Reader/Writer (XML und Text):

```
Altova.IO.ReaderInput(System.IO.TextReader reader)
Altova.IO.WriterOutput(System.IO.TextWriter writer)
```

DOM-Dokumente (nur XML):

```
Altova.IO.DocumentInput(System.Xml.XmlDocument document)
Altova.IO.DocumentOutput(System.Xml.XmlDocument document)
```

MapForce generiert die run-Funktion, die mehrere **Inputs** und einen **Output** erhält.

**Einfaches Beispiel**

Wir verwenden eine Java (oder C#) Applikation und möchten, dass MapForce Mapping-Code generiert, der in unsere Applikation integriert werden soll. Beispiel: Das Mapping besteht aus zwei XML-Quelldateien und einer Ziel-Textdatei. MapForce generiert die folgende run-Funktion:

```
void run(Input in1, Input in2, Output out1);
```

Angenommen mit der Applikation ist es erforderlich, dass wir die Daten von einer lokalen Datei und einem Binärstream auf einen Zeichenstream mappen. Die Daten stammen aus anderen Quellen und wir möchten den von MapForce generierten Code in unsere Applikation integrieren. Die Applikation stellt die Quellen und Ziele folgendermaßen zur Verfügung:

```
String filename;
Java.io.InputStream stream;
Java.io.Writer writer;
```

Für die von MapForce generierte run-Funktion müssen die folgenden Wrapper erzeugt werden:

```
// com.altova.io is considered imported here:
Input input1 = new FileInput(filename);
Input input2 = new StreamInput(stream);
Output output1 = new WriterOutput(writer);
```

An dieser Stelle muss die von MapForce generierte run-Funktion aufgerufen werden:

```
MappingMapToSomething.run(input1, input2, output1);
```

Fertig.

Das Verhalten bei **C#** ist beinahe identisch, mit der Ausnahme, dass "run" **Run** heißt und der .NET-Stream und der Reader/Writer in C# anders heißen.

Andere Input / Output-Typen wie z.B. Strings oder DOM-Dokumente werden auf dieselbe Art

verwendet. Die Unterschiede sehen Sie unten.

### Verhalten von Streams

Beachten Sie bei Binärdateien oder Zeichenstreams die folgenden Regeln:

- Streams und Readers/Writers müssen geöffnet und verwendungsbereit sein, bevor run aufgerufen wird.
- Standardmäßig schließt die von MapForce generierte run-Funktion Streams, Readers/Writers, wenn der Vorgang abgeschlossen ist. Wenn dies unerwünscht ist, fügen Sie vor dem Aufruf der run-Funktion folgenden Code ein:

```
MappingMapToSomething.setCloseObjectsAfterRun(true); // Java
```

oder

```
MappingMapToSomething.CloseObjectsAfterRun = false; // C#
```

- Excel-Quellen/Ziele können aus Zeichenstreams nicht gelesen werden und nicht in Zeichenstreams geschrieben werden.

### Verhalten von Strings

Zwischen dem Verhalten des C#- und Java StringOutput-Verhaltens gibt es einen kleinen Unterschied.

In **Java** erhält StringOutput kein Argument. Inhalt kann folgendermaßen abgerufen werden:

```
// mapping from String to (another) String

String blah = "<here>is some xml text</here>";

Input input = new StringInput(blah);
Output output = new StringOutput(); // used where?

MappingMapToBlah.run(input, output);

String myTargetData = output.getString().toString(); // here!
```

Die **getString()** Methode gibt ein *StringBuffer* zurück. Daher wird auch **toString()** benötigt.

In **C#** erhält StringOutput ein Argument (StringBuilder), das Sie vorher bereitstellen müssen. Der StringBuilder kann bereits vorher Daten enthalten, daher wird die Mapping-Ausgabe daran angehängt.

- Excel-Quellen/Ziele können nicht von/auf Strings gemappt werden.

### Verhalten von DOM-Dokumenten

Falls dies in Ihrer Applikation erforderlich ist, können Sie DocumentInput / DocumentOutput als Argumente an "run" übergeben.

Beachten Sie, dass das an den DocumentOutput-Konstruktor übergebene Dokument im Zielmodus leer sein muss.

Nach Aufruf von "run" enthält das vom Konstruktor von DocumentOutput bereitgestellte/generierte DOM-Dokument bereits gemappte Daten, daher ist ein "speichere in Dokument" nicht erforderlich. Nach dem Mapping kann das Dokument nach Ihren Wünschen bearbeitet werden.

Nur XML-Inhalt kann auf DOM-Dokumente gemappt werden.

### Datenbanken

Wenn ein Mapping Datenbankkomponenten hat, enthält die run-Funktion an der entsprechenden Stelle das Datenbankverbindungsobjekt.

Wenn z.B. von Text-, XML- und Dateninhalt aus einer Datenbank auf eine Output-Datei gemappt wird, generiert MapForce die folgende run-Funktion:

```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output out1);
```

Die Reihenfolge der Argumente ist hier von Bedeutung. Sie können mit dbConn Parametern experimentieren oder beim Integrieren von Code die von MapForce generierten Standardparameter verwenden.



## 31.6 Verwendung der generierten Codebibliothek

### Übersicht

Bei der Codegenerierung wird in MapForce eine komplette Applikation generiert, die alle Mapping-Schritte automatisch ausführt. Wenn Sie planen, den Mapping-Code in Ihre eigene Applikation zu integrieren, können Sie Bibliotheken für alle im Mapping verwendeten Schemas generieren. Mit Hilfe dieser Bibliotheken kann Ihr Code jederzeit XML-Instanzen erstellen oder lesen, die vom Mapping-Code verwendet oder erstellt wurden.

Wenn Sie im [Dialogfeld "Optionen"](#) festlegen, dass Wrapper-Klassen generiert werden sollen, erstellt MapForce nicht nur die Mapping-Applikation für Sie, sondern erstellt zusätzlich dazu Wrapper-Klassen für alle im Mapping verwendeten Schemas. Diese können in Ihrem eigenen Code auf dieselbe Art verwendet werden wie vom XMLSpy Code Generator generierte Bibliotheken.

Bei der von MapForce generierten Bibliothek handelt es sich um einen DOM (W3C Document Object Model) Wrapper, mit dem Sie XML-Dokumente einfach und sicher lesen, ändern und erstellen können. Alle Daten werden im DOM zusammengefasst. Es gibt Methoden, um Daten aus dem DOM zu extrahieren und zu aktualisieren und Daten in das DOM aufzunehmen.

### Kompatibilitätsmodus

Je nach dem Wert in der Einstellung "Generiere Code kompatibel mit" haben die generierten Klassen unterschiedliche Namen, Member und Methoden. Die folgende Beschreibung gilt **nur für Versionen ab Version 2007r3**. Informationen zu den restlichen Einstellungen finden Sie im Kapitel [Verwendung von generiertem Code, der mit älteren Versionen kompatibel ist](#). Es gab seit Version 2007r3 keine Änderungen, die die Kompatibilität beeinträchtigen.

### Generierte Bibliotheken

Wenn MapForce Code anhand eines XML-Schemas oder einer DTD generiert, werden die folgenden Bibliotheken generiert.

| Name            | Verwendungszweck                                                                                                              |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------|
| Altova          | Basisbibliothek, enthält allgemeine Runtime-Unterstützung, ist für alle Schemas identisch                                     |
| AltovaXML       | Basisbibliothek, enthält Runtime-Unterstützung für XML, ist für jedes Schema identisch                                        |
| YourSchema      | Bibliothek mit Deklarationen, die anhand des Eingabeschemas generiert wurden. Wird wie die Schema-Datei bzw. die DTD benannt. |
| YourSchema Test | Testapplikationsgerüst (nur XMLSpy)                                                                                           |

### Namensgenerierung und Namespaces

Im Allgemeinen versucht der Code Generator die Namen für generierte Namespaces, Klassen

und Member aus dem Original-XML-Schema beizubehalten. Zeichen, die in Identifiern in der Zielsprache ungültig sind, werden durch ein "\_" ersetzt. Namen, die mit anderen Namen oder reservierten Wörtern in Konflikt treten würden, werden durch Anhängen einer Nummer eindeutig identifizierbar gemacht. Die Generierung von Namen kann durch Änderung der [Standardeinstellungen](#) in der SPL-Vorlage beeinflusst werden.

Die Namespaces aus dem XML-Schema werden in Java in Pakete oder in C#- oder C++-Code in Namespaces konvertiert, wobei das Namespace-Präfix aus dem Schema als Code-Namespace verwendet wird. Die gesamte Bibliothek wird in ein Paket oder einen anhand des Schema-Dateinamens abgeleiteten Namespace eingeschlossen, sodass Sie in einem Programm mehrere generierte Bibliotheken verwenden können, ohne dass es zu Namenskonflikten kommt.

### **Datentypen**

XML-Schema hat ein etwas komplexeres Datentypmodell als Java, C# oder C++. Der Code Generator konvertiert die 44 XML-Schematypen in einige sprachspezifische primitive Typen oder in mit der Altova-Bibliothek bereitgestellte Klassen. Die Zuordnung von simpleTypes kann in der SPL-Vorlage konfiguriert werden.

ComplexTypes und abgeleitete Typen, die im Schema definiert sind, werden in der generierten Hierarchie in Klassen konvertiert.

Enumeration Facets von simpleTypes werden in Symbolkonstanten konvertiert.

### **Generierte Klassen**

MapForce generiert im Allgemeinen eine Klasse pro Typ, der im XML-Schema gefunden wird, bzw. pro Element in der DTD. Eine weitere Klasse pro Bibliothek wird zur Darstellung des Dokuments selbst generiert. Sie dient zum Laden und Speichern des Dokuments.

### **Speicherverwaltung**

Eine DOM-Struktur besteht aus Nodes, die immer einem bestimmten DOM-Dokument zugeordnet sind - selbst, wenn die Nodes derzeit nicht im Inhalt des Dokuments vorkommen. Bei allen generierten Klassen handelt es sich um Referenzen auf die DOM-Nodes, für die sie stehen, nicht um Werte. Dies bedeutet, dass bei Zuweisung einer Instanz einer generierten Klasse nicht der Wert kopiert wird, sondern nur eine zusätzliche Referenz auf dieselben Daten.

### 31.6.1 Beispielschema

#### Verwendung der generierten Klassen

Der Einfachheit halber werden wir mit einer sehr kleinen XSD-Datei arbeiten. Unten sehen Sie den Quellcode dazu. Speichern Sie die Datei als "**Library.xsd**", wenn Sie dieselben Ergebnisse wie in diesem Beispiel erhalten wollen:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.nanonull.com/LibrarySample" xmlns:xs="
http://www.w3.org/2001/XMLSchema" targetNamespace="
http://www.nanonull.com/LibrarySample" elementFormDefault="qualified" attributeFormDefault
="unqualified">
 <xs:element name="Library">
 <xs:complexType>
 <xs:sequence>
 <xs:element name="Book" type="BookType" minOccurs="0"
maxOccurs="unbounded"/>
 </xs:sequence>
 </xs:complexType>
 </xs:element>
 <xs:complexType name="BookType">
 <xs:sequence>
 <xs:element name="Title" type="xs:string"/>
 <xs:element name="Author" type="xs:string" maxOccurs="unbounded
"/>
 </xs:sequence>
 <xs:attribute name="ISBN" type="xs:string" use="required"/>
 <xs:attribute name="Variant" type="BookVariant"/>
 </xs:complexType>
 <xs:simpleType name="BookVariant">
 <xs:restriction base="xs:string">
 <xs:enumeration value="Hardcover"/>
 <xs:enumeration value="Paperback"/>
 <xs:enumeration value="Audiobook"/>
 <xs:enumeration value="E-book"/>
 </xs:restriction>
 </xs:simpleType>
</xs:schema>
```

In diesem Beispiel werden wir nur diese Schema-Datei verwenden, damit man die Unterschiede zwischen den verschiedenen Programmiersprachen in den folgenden Beispielen leicht erkennt.

Beachten Sie bitte, dass die generierten Klassen von XSD-Datei zu XSD-Datei unterschiedlich benannt werden. Stellen Sie sicher, dass Sie die Namen der Funktionen und Variablen anpassen, wenn Sie mit Ihren eigenen Schemas oder anderen Beispielen arbeiten.

### 31.6.2 Verwendung der generierten Java-Bibliothek

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Generiere Code kompatibel mit V2007r3** im Dialogfeld "Codegenerierung" beschrieben.

#### Generierte Pakete

| Name               | Verwendungszweck                                                                                                        |
|--------------------|-------------------------------------------------------------------------------------------------------------------------|
| com.altova         | Basispaket mit der allgemeinen Runtime-Unterstützung, für alle Schemas identisch                                        |
| com.altova.xml     | Basispaket mit der allgemeinen Runtime-Unterstützung für XML, für alle Schemas identisch                                |
| com.YourSchema     | Paket mit Deklarationen, die anhand des Eingabeschemas generiert wurden. Wird wie die Schema-Datei bzw. die DTD benannt |
| com.YourSchemaTest | Testapplikationsgerüst (nur XMLSpy)                                                                                     |

#### DOM-Implementierung

Für den generierten Java-Code wird als zugrunde liegende DOM-Schnittstelle JAXP (Java API for XML Processing) verwendet. Bitte beachten Sie, dass die mit Java 1.4 bereitgestellte Standard-DOM-Implementierung beim Serialisieren von XML-Dokumenten nicht automatisch Namespace-Attribute erstellt. Wenn Sie auf Probleme stoßen, installieren Sie bitte ein Update auf Java 5 oder höher.

#### Datentypen

Für das Mappen von XML-Schematypen auf Java-Datentypen wird das folgende Standard-Mapping verwendet:

| XML-Schema                                                          | Java                      | Anmerkungen                                                         |
|---------------------------------------------------------------------|---------------------------|---------------------------------------------------------------------|
| xs:string                                                           | String                    |                                                                     |
| xs:boolean                                                          | boolean                   |                                                                     |
| xs:decimal                                                          | java.math.BigDecimal      |                                                                     |
| xs:float, xs:double                                                 | double                    |                                                                     |
| xs:integer                                                          | java.math.BigInteger      |                                                                     |
| xs:long                                                             | long                      |                                                                     |
| xs:unsignedLong                                                     | java.math.BigInteger      | Java hat keine nicht signierten Typen.                              |
| xs:int                                                              | int                       |                                                                     |
| xs:unsignedInt                                                      | long                      | Java hat keine nicht signierten Typen.                              |
| xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth | com.altova.types.DateTime |                                                                     |
| xs:duration                                                         | com.altova.types.Duration |                                                                     |
| xs:hexBinary and xs:base64Binary                                    | byte[]                    | Die Kodierung und Dekodierung von Binärdateien erfolgt automatisch. |
| xs:anySimpleType                                                    | string                    |                                                                     |

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben Java-Typ wie der entsprechende Basistyp gemappt.

### Generierte Dokumentklasse

Zusätzlich zu den Klassen für die Typen, die im XML-Schema deklariert wurden, wird die Dokumentklasse generiert. Sie enthält als Member alle möglichen Root-Elemente sowie zusätzlich dazu die folgenden Methoden ("Doc" steht für den Namen der generierten Dokumentklasse selbst):

| Methode                                                                                                             | Verwendungszweck                                                                                                                                                     |
|---------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| static <i>Doc</i> <b>loadFromFile</b> (String fileName)                                                             | Lädt ein XML-Dokument aus einer Datei                                                                                                                                |
| static <i>Doc</i> <b>loadFromString</b> (String xml)                                                                | Lädt ein XML-Dokument aus einem String                                                                                                                               |
| static <i>Doc</i> <b>loadFromBinary</b> (byte[] xml)                                                                | Lädt ein XML-Dokument aus einem Byte Array                                                                                                                           |
| void <b>saveToFile</b> (String fileName, boolean prettyPrint, String encoding = "UTF-8")                            | Speichert ein XML-Dokument in einer Datei mit der definierte Kodierung. Ist keine Kodierung definiert, wird UTF-8 verwendet.                                         |
| void <b>SaveToFile</b> (String fileName, boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM) | Speichert ein XML-Dokument in einer Datei mit der definierte Kodierung. Für Unicode-Kodierungen können Bytefolge und Unicode Byte-Markierung definiert werden.       |
| String <b>saveToString</b> (boolean prettyPrint)                                                                    | Speichert ein XML-Dokument in einem String                                                                                                                           |
| byte[] <b>saveToBinary</b> (boolean prettyPrint, String encoding = "UTF-8")                                         | Speichert ein XML-Dokument in einem Byte Array mit der angegebenen Kodierung. Wird keine Kodierung angegeben, wird UTF-8 verwendet.                                  |
| byte[] <b>saveToBinary</b> (boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)              | Speichert ein XML-Dokument in einem Byte Array mit der angegebenen Kodierung. Für Unicode-Kodierungen können Bytefolge und Unicode Byte-Markierung definiert werden. |
| static <i>Doc</i> <b>createDocument</b> ()                                                                          | Erstellt ein neues leeres XML-Dokument                                                                                                                               |
| void <b>setSchemaLocation</b> (String schemaLocation)                                                               | Fügt ein xsi:schemaLocation oder ein xsi:noNamespaceSchemaLocation Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein.                 |

### Generierte Typklasse

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für simple und mixed Types werden **getValue()** und **setValue(string)** Methoden generiert. Für simple types mit Enumeration Facets können **getEnumerationValue()** und **setEnumerationValue(int)** zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des Weiteren können Sie mit Hilfe der Methode **getStaticInfo()** Schemainformationen als **com.altova.xml.meta.SimpleType** oder **com.altova.xml.meta.ComplexType** (siehe unten) abrufen.

Durch die XML Schema-Ableitung von complexTypes nach Extension werden in der generierten Bibliothek abgeleitete Klassen erzeugt.

### Generierte Member Attribut-Klasse

Für jedes Member Attribut eines Typs wird eine Klasse mit den folgenden Methoden erstellt.

| Methode                                 | Verwendungszweck                                             |
|-----------------------------------------|--------------------------------------------------------------|
| boolean <b>exists</b> ()                | Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist |
| void <b>remove</b> ()                   | Entfernt das Attribut aus dem übergeordneten Element         |
| <i>AttributeType</i> <b>getValue</b> () | Ruft den Attributwert ab                                     |

|                                                      |                                                                                                                                                                                                             |
|------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>void setValue(AttributeType value)</code>      | Definiert den Attributwert                                                                                                                                                                                  |
| <code>int getEnumerationValue()</code>               | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| <code>void setEnumerationValue(int)</code>           | Wird nur für Enumeration Types generiert. Übergeben Sie eine der für die möglichen Werte generierten Konstanten an diese Methode, um den Wert zu definieren.                                                |
| <code>com.altova.xml.meta.Attribute getInfo()</code> | Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe unten)                                                                                                                                   |

### Generierte Member Element-Klasse

Für jedes Member Element eines Typs wird eine Klasse mit den folgenden Methoden erstellt.

| Methode                                            | Verwendungszweck                                                                                                                                                                                            |
|----------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>MemberType first()</code>                    | Gibt die erste Instanz des Member Elements zurück                                                                                                                                                           |
| <code>MemberType at(int index)</code>              | Gibt die Instanz des Member Elements zurück                                                                                                                                                                 |
| <code>MemberType last()</code>                     | Gibt die letzte Instanz des Member Elements zurück                                                                                                                                                          |
| <code>MemberType append()</code>                   | Erstellt ein neues Element und hängt es an das übergeordnete Element an                                                                                                                                     |
| <code>boolean exists()</code>                      | Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist                                                                                                                                      |
| <code>int count()</code>                           | Gibt die Anzahl der Elemente zurück                                                                                                                                                                         |
| <code>void remove()</code>                         | Löscht alle Instanzen des Elements aus dem übergeordneten Element                                                                                                                                           |
| <code>void removeAt(int index)</code>              | Löscht die Instanz des durch den Index definierten Elements                                                                                                                                                 |
| <code>java.util.Iterator iterator()</code>         | Gibt ein Objekt für iterierende Instanzen des Member Elements zurück                                                                                                                                        |
| <code>MemberType getValue()</code>                 | Ruft den Elementinhalt ab (wird nur generiert, wenn das Element simple oder mixed Content haben kann)                                                                                                       |
| <code>void setValue(MemberType value)</code>       | Definiert den Elementinhalt (wird nur generiert, wenn das Element simple oder mixed Content haben kann)                                                                                                     |
| <code>int getEnumerationValue()</code>             | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| <code>void setEnumerationValue(int)</code>         | Wird nur für Enumeration Types generiert. Übergeben Sie eine der für die möglichen Werte generierten Konstanten an diese Methode, um den Wert zu definieren.                                                |
| <code>com.altova.xml.meta.Element getInfo()</code> | Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe unten)                                                                                                                                   |

### Erstellen von XML-Dokumenten

Mit der generierten Codebibliothek wird die Erstellung von XML-Dateien sehr einfach. Beginnen

wir mit einem XML-Beispieldatei für das [Beispielschema](#):

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" xmlns="
http://www.nanonull.com/LibrarySample" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
 <Book ISBN="0764549642" Variant="Paperback">
 <Title>The XML Spy Handbook</Title>
 <Author>Altova</Author>
 </Book>
</Library>
```

Zur Erstellung dieser Datei wurde der folgende Java-Code verwendet. Die Klassen Library2, LibraryType und BookType werden anhand des [Schemas](#) generiert und stellen das gesamte Dokument, den Typ des <Library> Elements und den complexType BookType dar, der der Typ des <Book> Elements ist. Beachten Sie, dass die Funktion zur automatischen Vermeidung von Namenskonflikten die Bibliotheksklasse in Library2 umbenannt hat, um eventuelle Konflikte mit dem Namespace-Namen der Bibliothek zu vermeiden.

```
protected static void example() throws Exception {
 // create a new, empty XML document
 Library2 libDoc = Library2.createDocument();

 // create the root element <Library> and add it to the document
 LibraryType lib = libDoc.Library3.append();

 // create a new <Book> and add it to the library
 BookType book = lib.Book.append();

 // set the ISBN attribute of the book
 book.ISBN.setValue("0764549642");

 // set the Variant attribute of the book using an enumeration
 constant book.Variant.setEnumerationValue(BookVariant.EPAPERBACK);

 // add the <Title> and <Author> elements, and set values
 book.Title.append().setValue("The XML Spy Handbook");
 book.Author.append().setValue("Altova");

 // set the schema location (this is optional)
 libDoc.setSchemaLocation("Library.xsd");

 // save the XML document to a file with default encoding (UTF-8)
 // "true" causes the file to be pretty-printed.
 libDoc.saveToFile("Library1.xml", true);
}
```

Beachten Sie, dass alle Elemente durch Aufruf von append() erstellt werden und dass Attribute (wie in diesem Beispiel ISBN) ganz einfach wie Struktur-Member aufgerufen werden können.

### Lesen von XML-Dokumenten

Im folgenden Beispiel wird die soeben erstellte Datei gelesen. Es werden die verschiedenen Methoden zum Lesen von XML-Dateien gezeigt:

```
protected static void example() throws Exception {
 // load XML document
 Library2 libDoc = Library2.loadFromFile("Library1.xml");

 // get the first (and only) root element <Library>
 LibraryType lib = libDoc.Library3.first();
}
```

```

// it is possible to check whether an element exists:
if (!lib.Book.exists())
{
 System.out.println("This library is empty.");
 return;
}

// iteration: for each <Book>...
for (java.util.Iterator itBook = lib.Book.iterator();
itBook.hasNext();)
{
 BookType book = (BookType) itBook.next();

 // output values of ISBN attribute and (first and only)
 System.out.println("ISBN: " + book.ISBN.getValue());
 System.out.println("Title: " + book.Title.first().get
Value());

 // read and compare an enumeration value
 if (book.Variant.getEnumerationValue() ==
BookVariant.EPAPERBACK)
 System.out.println("This is a paperback book.");

 // for each <Author>...
 for (java.util.Iterator itAuthor = book.Author.iterator();
itAuthor.hasNext();)
 System.out.println("Author: " +
((com.Library.xs.stringType)itAuthor.next()).getValue());

 // alternative: use count and index
 for (int j = 0; j < book.Author.count(); ++j)
 System.out.println("Author: " + book.Author.at(j)
.getValue());
}
}

```

Beachten Sie: Der Typ des book.Author Member: xs.stringType ist eine generierte Klasse für jedes Element mit einfachem String-Inhalt. Ähnliche Klassen werden für alle anderen XML-Schematypen generiert.

Sie können natürlich auch ein Element laden, es durch Hinzufügen oder Löschen von Elementen und Attributen ändern und die geänderte Datei anschließend wieder speichern.

### Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace com.altova sind die folgenden Ausnahmeereignisklassen definiert:

| Klasse                             | Basisklasse | Beschreibung                                                  |
|------------------------------------|-------------|---------------------------------------------------------------|
| SourceInstanceUnavailableException | Exception   | Beim Laden einer XML-Instanz ist ein Problem aufgetreten.     |
| TargetInstanceUnavailableException | Exception   | Beim Speichern einer XML-Instanz ist ein Problem aufgetreten. |

Zusätzlich dazu werden die folgenden Java-Ausnahmeereignisse verwendet:

| Klasse          | Beschreibung                                                           |
|-----------------|------------------------------------------------------------------------|
| java.lang.Error | Interner Fehler in der Programmlogik (unabhängig von den Eingabedaten) |



|                                    |                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| java.lang.Exception                | Basisklasse für Runtime-Fehler                                                                                       |
| java.lang.IllegalArgumentException | Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen, oder es gab einen Fehler bei einer Typkonvertierung. |
| java.lang.ArithmeticException      | Ausnahmeereignis aufgrund eines Fehlers bei der Konvertierung eines numerischen Typs.                                |

### Aufrufen von Metadaten

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden. Die Methoden, die eine der Metadatenklassen zurückgeben, geben Null zurück, wenn die entsprechende Eigenschaft nicht existiert.

#### com.altova.xml.meta.SimpleType

| Methode                           | Beschreibung                                                                                                                                                                                                                                                                                                  |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SimpleType <b>getBaseType()</b>   | Gibt den Base Type dieses Typs zurück                                                                                                                                                                                                                                                                         |
| String <b>getLocalName()</b>      | Gibt den lokalen Namen des Typs zurück                                                                                                                                                                                                                                                                        |
| String <b>getNamespaceURI()</b>   | Gibt die Namespace URI des Typs zurück                                                                                                                                                                                                                                                                        |
| int <b>getMinLength()</b>         | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| int <b>getMaxLength()</b>         | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| int <b>getLength()</b>            | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| int <b>getTotalDigits()</b>       | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| int <b>getFractionDigits()</b>    | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| String <b>getMinInclusive()</b>   | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| String <b>getMinExclusive()</b>   | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| String <b>getMaxInclusive()</b>   | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| String <b>getMaxExclusive()</b>   | Gibt den Wert dieses Facet zurück                                                                                                                                                                                                                                                                             |
| String[] <b>getEnumerations()</b> | Gibt ein Array aller Enumeration Facets zurück                                                                                                                                                                                                                                                                |
| String[] <b>getPatterns()</b>     | Gibt ein Array aller Pattern Facets zurück                                                                                                                                                                                                                                                                    |
| int <b>getWhitespace()</b>        | Gibt den Wert des Whitespace Facet zurück, der einer der folgenden ist:<br>com.altova.typeinfo.WhitespaceType.Whitespace_Unknown<br>com.altova.typeinfo.WhitespaceType.Whitespace_Preserve<br>com.altova.typeinfo.WhitespaceType.Whitespace_Replace<br>com.altova.typeinfo.WhitespaceType.Whitespace_Collapse |

#### com.altova.xml.meta.ComplexType

| Methode                                                                | Beschreibung                                                               |
|------------------------------------------------------------------------|----------------------------------------------------------------------------|
| Attribute[] <b>GetAttributes()</b>                                     | Gibt eine Liste aller Attribute zurück                                     |
| Element[] <b>GetElements()</b>                                         | Gibt eine Liste aller Elemente zurück                                      |
| ComplexType <b>getBaseType()</b>                                       | Gibt den Base Type dieses Typs zurück                                      |
| String <b>getLocalName()</b>                                           | Gibt den lokalen Namen dieses Typs zurück                                  |
| String <b>getNamespaceURI()</b>                                        | Gibt die Namespace URI des Typs zurück                                     |
| SimpleType <b>getContentTyp()</b>                                      | Gibt den simpleType des Inhalts zurück                                     |
| Element <b>findElement</b> (String localName, String namespaceURI)     | Sucht das Element mit dem definierten lokalen Namen und der Namespace URI  |
| Attribute <b>findAttribute</b> (String localName, String namespaceURI) | Sucht das Attribut mit dem definierten lokalen Namen und der Namespace URI |

#### com.altova.xml.meta.Element

| Methode                          | Beschreibung                                                                                                                                                                                                                                              |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ComplexType <b>getDataType()</b> | Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie getContentTypes() des zurückgegebenen Objekts zum den simple content type abzurufen. |
| int <b>getMinOccurs()</b>        | Gibt den im Schema definierten minOccurs Wert zurück                                                                                                                                                                                                      |
| int <b>getMaxOccurs()</b>        | Gibt den im Schema definierten maxOccurs Wert zurück                                                                                                                                                                                                      |
| String <b>getLocalName()</b>     | Gibt den lokalen Namen des Elements zurück                                                                                                                                                                                                                |
| String <b>getNamespaceURI()</b>  | Gibt die Namespace URI des Elements zurück                                                                                                                                                                                                                |

**com.altova.xml.meta.Attribute**

| Methode                         | Beschreibung                                           |
|---------------------------------|--------------------------------------------------------|
| SimpleType <b>getDataType()</b> | Gibt den Typ des Attributinhalt zurück                 |
| boolean <b>isRequired()</b>     | Gibt "true" zurück, wenn das Attribut erforderlich ist |
| String <b>getLocalName()</b>    | Gibt den lokalen Namen des Attributs zurück            |
| String <b>getNamespaceURI()</b> | Gibt die Namespace URI des Attributs zurück            |

### 31.6.3 Verwendung der generierten C++-Bibliothek

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Generiere Code kompatibel mit V2007r3** im Dialogfeld "Codegenerierung" beschrieben.

#### Generierte Bibliotheken

| Name            | Verwendungszweck                                                                                                             |
|-----------------|------------------------------------------------------------------------------------------------------------------------------|
| Altova          | Basisbibliothek mit der allgemeinen Runtime-Unterstützung, für alle Schemas identisch                                        |
| AltovaXML       | Basisbibliothek mit der allgemeinen Runtime-Unterstützung für XML, für alle Schemas identisch                                |
| YourSchema      | Bibliothek mit Deklarationen, die anhand des Eingabeschemas generiert wurden. Wird wie die Schema-Datei bzw. die DTD benannt |
| YourSchema Test | Testapplikationsgerüst (nur XMLSpy)                                                                                          |

#### DOM-Implementierungen

Der generierte C++-Code unterstützt, je nach Codegenerierungseinstellungen, entweder Microsoft MSXML oder Apache Xerces 2.6Mi oder höher. Die Syntax zur Verwendung des generierten Codes ist bei beiden DOM-Implementierungen dieselbe.

#### Zeichentypen

Der generierte C++-Code kann mit oder ohne Unicode-Unterstützung generiert werden. Je nachdem, welche Einstellung hier gewählt wurde, werden die Typen **string\_type** und **tstring** beide als **std::string** und **std::wstring** definiert und zwar bestehend aus schmalen oder breiten Zeichen, definiert. Um Unicode-Zeichen in Ihrer XML-Datei verwenden zu können, die nicht durch den aktuellen 8-Bit-Zeichensatz dargestellt werden, muss die Unicode-Unterstützung aktiviert werden. Seien Sie bei den **\_T()** Makros vorsichtig. Dieses Makro stellt sicher, dass String-Konstanten korrekt gespeichert werden, ob Sie nun für Unicode oder Nicht-Unicode-Programme kompilieren.

#### Datentypen

Für das Mappen von XML-Schematypen auf C++-Datentypen wird das folgende Standard-Mapping verwendet:

| XML Schema                                                          | C++              | Anmerkungen                                                                                   |
|---------------------------------------------------------------------|------------------|-----------------------------------------------------------------------------------------------|
| xs:string                                                           | string_type      | string_type wird definiert als std::string oder std::wstring                                  |
| xs:boolean                                                          | bool             |                                                                                               |
| xs:decimal                                                          | double           | C++ hat keinen Dezimaltyp, daher wird "double" verwendet.                                     |
| xs:float, xs:double                                                 | double           |                                                                                               |
| xs:integer                                                          | _int64           | xs:integer hat keine Bereichsbeschränkung, wird aus Gründen der Effizienz auf _int64 gemappt. |
| xs:nonNegativeInteger                                               | unsigned int64   | siehe oben                                                                                    |
| xs:int                                                              | int              |                                                                                               |
| xs:unsignedInt                                                      | unsigned int     |                                                                                               |
| xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth | altova::DateTime |                                                                                               |
| xs:duration                                                         | altova::Duration |                                                                                               |

|                                     |                            |                                                                        |
|-------------------------------------|----------------------------|------------------------------------------------------------------------|
| xs:hexBinary and<br>xs:base64Binary | std::vector<unsigned char> | Die Kodierung und Dekodierung von<br>Binärdateien erfolgt automatisch. |
| xs:anySimpleType                    | string_type                |                                                                        |

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben C++-Typ wie der entsprechende Basistyp gemappt.

### Generierte Dokumentklasse

Zusätzlich zu den Klassen für die Typen, die im XML-Schema deklariert wurden, wird die Dokumentklasse generiert. Sie enthält als Member alle möglichen Root-Elemente sowie zusätzlich dazu die folgenden Methoden ("Doc" steht für den Namen der generierten Dokumentklasse selbst):

| Methode                                                                                                                            | Verwendungszweck                                                                                                                                                                                                                                                         |
|------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| static <i>CDoc</i> <b>LoadFromFile</b> (const string_type& fileName)                                                               | Lädt ein XML-Dokument aus einer Datei                                                                                                                                                                                                                                    |
| static <i>CDoc</i> <b>LoadFromString</b> (const string_type& xml)                                                                  | Lädt ein XML-Dokument aus einem String                                                                                                                                                                                                                                   |
| static <i>CDoc</i> <b>LoadFromBinary</b> (const std::vector<unsigned char>& xml)                                                   | Lädt ein XML-Dokument aus einem Byte Array                                                                                                                                                                                                                               |
| void <b>SaveToFile</b> (const string_type& fileName, bool prettyPrint, const string_type& encoding = "UTF-8")                      | Speichert ein XML-Dokument in einer Datei mit der definierten Kodierung. Ist keine Kodierung definiert, wird UTF-8 verwendet.                                                                                                                                            |
| void <b>SaveToFile</b> (const string_type& fileName, bool prettyPrint, const string_type& encoding, bool bigEndian, bool writeBOM) | Speichert ein XML-Dokument in einer Datei mit der definierte Kodierung. Für Unicode-Kodierungen können Bytefolge und Unicode Byte-Markierung definiert werden.                                                                                                           |
| string_type <b>SaveToString</b> (bool prettyPrint)                                                                                 | Speichert ein XML-Dokument in einem String                                                                                                                                                                                                                               |
| std::vector<unsigned char> <b>SaveToBinary</b> (bool prettyPrint, const string_type& encoding = "UTF-8")                           | Speichert ein XML-Dokument in einem Byte Array mit der angegebenen Kodierung. Wird keine Kodierung angegeben, wird UTF-8 verwendet                                                                                                                                       |
| std::vector<unsigned char> <b>SaveToBinary</b> (bool prettyPrint, const string_type& encoding, bool bigEndian, bool writeBOM)      | Speichert ein XML-Dokument in einem Byte Array mit der angegebenen Kodierung. Für Unicode-Kodierungen können Bytefolge und Unicode Byte-Markierung definiert werden.                                                                                                     |
| static <i>CDoc</i> <b>CreateDocument</b> ()                                                                                        | Erstellt ein neues leeres XML-Dokument. Muss mit <b>DestroyDocument</b> () freigegeben werden.                                                                                                                                                                           |
| void <b>DestroyDocument</b> ()                                                                                                     | Löscht das Dokument. Alle Referenzen auf das Dokument und seine Nodes werden ungültig gemacht. Dieses Methode muss aufgerufen werden, wenn Sie mit einem Dokument fertig sind.                                                                                           |
| void <b>SetSchemaLocation</b> (const string_type& schemaLocation)                                                                  | Fügt ein xsi:schemaLocation oder ein xsi:noNamespaceSchemaLocation Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein.                                                                                                                     |
| void <b>SetDTDLocation</b> (const string_type& dtdLocation)                                                                        | Fügt eine DOCTYPE-Deklaration mit der definierten System-ID hinzu. Es muss bereits ein Root-Element vorhanden sein. Diese Methode wird bei MSXML nicht unterstützt, da es nicht möglich ist, eine DOCTYPE-Deklaration zu einem Dokument im Arbeitsspeicher hinzuzufügen. |

### Generierte Typklasse

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für simple und mixed Types werden Zuweisungs- und Konvertierungsoperatoren generiert. Für simple types mit Enumeration Facets können **getEnumerationValue()** und **setEnumerationValue(int)** zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des weiteren können Sie mit Hilfe der Methode **StaticInfo()** Schemainformationen als **altova::meta::SimpleType** oder **altova::meta::ComplexType** (siehe unten) abrufen.

Durch die XML Schema-Ableitung von complexTypes nach Extension werden in der generierten Bibliothek abgeleitete Klassen erzeugt.

### Generierte Member Attribut-Klasse

Für jedes Member Attribut eines Typs wird eine Klasse mit den folgenden Methoden erstellt.

| Methode                               | Verwendungszweck                                                                                                                                                                                            |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| boolean <b>exists()</b>               | Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist                                                                                                                                                |
| void <b>remove()</b>                  | Entfernt das Attribut aus dem übergeordneten Element                                                                                                                                                        |
| int <b>getEnumerationValue()</b>      | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| void <b>SetEnumerationValue(int)</b>  | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| altova::meta::Attribute <b>info()</b> | Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe unten)                                                                                                                                   |

### Generierte Member Element-Klasse

Für jedes Member Element eines Typs wird eine Klasse mit den folgenden Methoden erstellt.

| Methode                                                  | Verwendungszweck                                                        |
|----------------------------------------------------------|-------------------------------------------------------------------------|
| <i>MemberType</i> <b>first()</b>                         | Gibt die erste Instanz des Member Elements zurück                       |
| <i>MemberType</i> <b>operator[]</b> (unsigned int index) | Gibt das vom Index definierte Member Element zurück                     |
| <i>MemberType</i> <b>last()</b>                          | Gibt die letzte Instanz des Member Elements zurück                      |
| <i>MemberType</i> <b>append()</b>                        | Erstellt ein neues Element und hängt es an das übergeordnete Element an |
| bool <b>exists()</b>                                     | Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist  |
| unsigned int <b>count()</b>                              | Gibt die Anzahl der Elemente zurück                                     |
| void <b>remove()</b>                                     | Löscht alle Instanzen des Elements aus dem übergeordneten Element       |
| void <b>remove</b> (unsigned int index)                  | Löscht die Instanz des durch den Index definierten Elements             |

|                                       |                                                                                                                                                                                                             |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Iterator<MemberType> <b>all</b> ()    | Gibt ein Objekt für iterierende Instanzen des Member Elements zurück                                                                                                                                        |
| int <b>GetEnumerationValue</b> ()     | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| void <b>SetEnumerationValue</b> (int) | Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| altova::meta::Element <b>info</b> ()  | Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe unten)                                                                                                                                   |

Bei simple und mixed Content Elementen werden Zuweisungs- und Konvertierungsoperatoren vom/zum native Type des Elements erstellt, damit es direkt in Zuweisungen verwendet werden kann.

### Erstellen von XML-Dokumenten

Mit der generierten Codebibliothek wird die Erstellung von XML-Dateien sehr einfach. Beginnen wir mit einem XML-Beispieldatei für das [Beispielschema](#):

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" xmlns="
http://www.nanonull.com/LibrarySample" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
 <Book ISBN="0764549642" Variant="Paperback">
 <Title>The XML Spy Handbook</Title>
 <Author>Altova</Author>
 </Book>
</Library>
```

Zur Erstellung dieser Datei wurde der folgende C++-Code verwendet. Die Klassen CLibrary, CLibraryType und CBookType werden anhand des [Schemas](#) generiert und stellen das gesamte Dokument, den Typ des <Library> Elements und den complexType BookType dar, der der Typ des <Book> Elements ist.

```
using namespace Library;
void Example()
{
 // create a new, empty XML document
 CLibrary libDoc = CLibrary::CreateDocument();

 // create the root element <Library> and add it to the document
 CLibraryType lib = libDoc.Library2.append();

 // create a new <Book> and add it to the library
 CBookType book = lib.Book.append();

 // set the ISBN attribute of the book
 book.ISBN = _T("0764549642");

 // set the Variant attribute of the book using an enumeration constant
 book.Variant.SetEnumerationValue(CBookVariant::k_Paperback);

 // add the <Title> and <Author> elements, and set values
 book.Title.append() = _T("The XML Spy Handbook");
```

```

book.Author.append() = _T("Altova");

// set the schema location (this is optional)
libDoc.SetSchemaLocation(_T("Library.xsd"));

// save the XML document to a file with default encoding (UTF-8).
// "true" causes the file to be pretty-printed.
libDoc.SaveToFile(_T("Library1.xml"), true);

// destroy the document - all references to the document and any of its
nodes become invalid
libDoc.DestroyDocument();
}

```

Beachten Sie, dass alle Elemente durch Aufruf von `append()` erstellt werden und dass Attribute (wie in diesem Beispiel ISBN) ganz einfach wie Struktur-Member aufgerufen werden können. Vergessen Sie nicht, das Dokument mit "Destroy" zu löschen, wenn Sie es fertig bearbeitet haben.

### Lesen von XML-Dokumenten

Im folgenden Beispiel wird die soeben erstellte Datei gelesen. Es werden die verschiedenen Methoden zum Lesen von XML-Dateien gezeigt:

```

using namespace Library;
void Example()
{
 // load XML document
 CLibrary libDoc = CLibrary::LoadFromFile(_T("Library1.xml"));

 // get the first (and only) root element <Library>
 CLibraryType lib = libDoc.Library2.first();

 // it is possible to check whether an element exists:
 if (!lib.Book.exists())
 {
 tcout << "This library is empty." << std::endl;
 return;
 }

 // iteration: for each <Book>...
 for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
 {
 // output values of ISBN attribute and (first and only) title
 tcout << "ISBN: " << tstring(itBook->ISBN) << std::endl;
 tcout << "Title: " << tstring(itBook->Title.first()) << std::endl;

 // read and compare an enumeration value
 if (itBook->Variant.GetEnumerationValue() ==
 CBookVariant::k_Paperback)
 tcout << "This is a paperback book." << std::endl;

 // for each <Author>...
 for (CBookType::Author::iterator itAuthor = itBook->Author.all(); itAuthor; ++itAuthor)
 tcout << "Author: " << tstring(itAuthor) << std::endl;

 // alternative: use count and index
 for (unsigned int j = 0; j < itBook->Author.count(); ++j)
 tcout << "Author: " << tstring(itBook->Author[j]) << std::endl;
 }

 // destroy the document - all references to the document and any of its
}

```

```

nodes become invalid
 LibDoc.DestroyDocument();
}

```

Um den Inhalt von Attributen und Elementen als Strings für den count << Operator aufzurufen, ist ein explizites "cast to tstring" erforderlich.

Sie können natürlich auch ein Element laden, es durch Hinzufügen oder Löschen von Elementen und Attributen ändern und die geänderte Datei anschließend wieder speichern.

### Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace `altova` sind die folgenden Ausnahmeereignisklassen definiert:

| Klasse                         | Basisklasse                     | Beschreibung                                                                           |
|--------------------------------|---------------------------------|----------------------------------------------------------------------------------------|
| Error                          | <code>std::logic_error</code>   | Interner Fehler in der Programmlogik (unabhängig von den Eingabedaten)                 |
| Exception                      | <code>std::runtime_error</code> | Basisklasse für Runtime-Fehler                                                         |
| InvalidArgumentsException      | Exception                       | Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen.                        |
| ConversionException            | Exception                       | Ausnahmeereignis aufgrund eines Typkonvertierungsfehlers                               |
| StringParseException           | ConversionException             | Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden. |
| ValueNotRepresentableException | ConversionException             | Ein Wert im Wertebereich kann nicht in einen lexikalischen Bereich umgewandelt werden. |
| OutOfRangeException            | ConversionException             | Ein Quellwert kann in der Ziel-Domain nicht dargestellt werden.                        |
| InvalidOperationException      | Exception                       | Es wurde versucht, eine Operation auszuführen, die in diesem Kontext nicht gültig ist. |
| DataSourceUnavailableException | Exception                       | Beim Laden einer XML-Instanz ist ein Problem aufgetreten.                              |
| DataTargetUnavailableException | Exception                       | Beim Speichern einer XML-Instanz ist ein Problem aufgetreten.                          |

Alle Ausnahmeereignisklassen enthalten einen Meldungstext und einen Verweis auf ein mögliches internes Ausnahmeereignis.

| Methode                             | Verwendungszweck                                                                                                                            |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <code>string_type message()</code>  | Gibt eine Textbeschreibung des Ausnahmeereignisses zurück                                                                                   |
| <code>std::exception inner()</code> | Gibt, falls verfügbar, das Ausnahmeereignis zurück, das dieses Ausnahmeereignis verursacht hat, ansonsten wird der Wert NULL zurückgegeben. |

### Aufrufen von Metadaten

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden. Alle Methoden werden als `const` deklariert. Die Methoden, die eine der Metadatenklassen zurückgeben, geben ein NULL-Objekt zurück, wenn die entsprechende Eigenschaft nicht existiert.

#### `altova::meta::SimpleType`

| Methode | Beschreibung |
|---------|--------------|
|---------|--------------|



|                                                   |                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>operator bool()</b>                            | Gibt "true" zurück, wenn dies nicht der NULL SimpleType ist                                                                                                       |
| <b>operator !()</b>                               | Gibt "true" zurück, wenn dies der NULL SimpleType ist                                                                                                             |
| SimpleType <b>GetBaseType()</b>                   | Gibt den Base Type dieses Typs zurück                                                                                                                             |
| string_type <b>GetLocalName()</b>                 | Gibt den lokalen Namen des Typs zurück                                                                                                                            |
| string_type <b>GetNamespaceURI()</b>              | Gibt die Namespace URI des Typs zurück                                                                                                                            |
| unsigned int <b>GetMinLength()</b>                | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| unsigned int <b>GetMaxLength()</b>                | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| unsigned int <b>GetLength()</b>                   | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| unsigned int <b>GetTotalDigits()</b>              | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| unsigned int <b>GetFractionDigits()</b>           | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| string_type <b>GetMinInclusive()</b>              | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| string_type <b>GetMinExclusive()</b>              | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| string_type <b>GetMaxInclusive()</b>              | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| string_type <b>GetMaxExclusive()</b>              | Gibt den Wert dieses Facet zurück                                                                                                                                 |
| std::vector<string_type> <b>GetEnumerations()</b> | Gibt eine Liste aller Enumeration Facets zurück                                                                                                                   |
| std::vector<string_type> <b>GetPatterns()</b>     | Gibt eine Liste aller Pattern Facets zurück                                                                                                                       |
| WhitespaceType <b>GetWhitespace()</b>             | Gibt den Wert des Whitespace Facet zurück, der einer der folgenden ist:<br>Whitespace_Unknown<br>Whitespace_Preserve<br>Whitespace_Replace<br>Whitespace_Collapse |

**altova::meta::ComplexType**

| Methode                                                                                    | Beschreibung                                                               |
|--------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| <b>operator bool()</b>                                                                     | Gibt "true" zurück, wenn dies nicht der NULL ComplexType ist               |
| <b>operator !()</b>                                                                        | Gibt "true" zurück, wenn dies der NULL ComplexType ist                     |
| std::vector<Attribute> <b>GetAttributes()</b>                                              | Gibt eine Liste aller Attribute zurück                                     |
| std::vector<Element> <b>GetElements()</b>                                                  | Gibt eine Liste aller Elemente zurück                                      |
| ComplexType <b>GetBaseType()</b>                                                           | Gibt den Base Type dieses Typs zurück                                      |
| string_type <b>GetLocalName()</b>                                                          | Gibt den lokalen Namen des Typs zurück                                     |
| string_type <b>GetNamespaceURI()</b>                                                       | Gibt die Namespace URI des Typs zurück                                     |
| SimpleType <b>GetContentType()</b>                                                         | Gibt dem simpleType des Inhalts zurück                                     |
| Element <b>FindElement</b> (const char_type* localName, const char_type* namespaceURI)     | Sucht das Element mit dem definierten lokalen Namen und der Namespace URI  |
| Attribute <b>FindAttribute</b> (const char_type* localName, const char_type* namespaceURI) | Sucht das Attribut mit dem definierten lokalen Namen und der Namespace URI |

**altova::meta::Element**

| Methode                | Beschreibung                                             |
|------------------------|----------------------------------------------------------|
| <b>operator bool()</b> | Gibt "true" zurück, wenn dies nicht das NULL Element ist |
| <b>operator !()</b>    | Gibt "true" zurück, wenn dies das NULL Element ist       |

|                                      |                                                                                                                                                                                                                                                          |
|--------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ComplexType <b>GetDataType()</b>     | Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie GetContentType() des zurückgegebenen Objekts, um den simple content type abzurufen. |
| unsigned int <b>GetMinOccurs()</b>   | Gibt den im Schema definierten minOccurs Wert zurück                                                                                                                                                                                                     |
| unsigned int <b>GetMaxOccurs()</b>   | Gibt den im Schema definierten maxOccurs Wert zurück                                                                                                                                                                                                     |
| string_type <b>GetLocalName()</b>    | Gibt den lokalen Namen des Elements zurück                                                                                                                                                                                                               |
| string_type <b>GetNamespaceURI()</b> | Gibt die Namespace URI des Elements zurück                                                                                                                                                                                                               |

**altova::meta::Attribute**

| Methode                              | Beschreibung                                              |
|--------------------------------------|-----------------------------------------------------------|
| operator bool()                      | Gibt "true" zurück, wenn dies nicht das NULL Attribut ist |
| operator !()                         | Gibt "true" zurück, wenn dies das NULL Attribut ist       |
| SimpleType <b>GetDataType()</b>      | Gibt den Typ des Attributinhalts zurück                   |
| bool <b>IsRequired()</b>             | Gibt "true" zurück, wenn das Attribut erforderlich ist    |
| string_type <b>GetLocalName()</b>    | Gibt den lokalen Namen des Attributs zurück               |
| string_type <b>GetNamespaceURI()</b> | Gibt die Namespace URI des Attributs zurück               |

### 31.6.4 Verwendung der generierten C#-Bibliothek

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Generiere Code kompatibel mit V2007r3** im Dialogfeld "Codegenerierung" beschrieben.

#### Generierte Bibliotheken

| Name            | Verwendungszweck                                                                                                             |
|-----------------|------------------------------------------------------------------------------------------------------------------------------|
| Altova          | Basisbibliothek mit der allgemeinen Runtime-Unterstützung, für alle Schemas identisch                                        |
| AltovaXML       | Basisbibliothek mit der allgemeinen Runtime-Unterstützung für XML, für alle Schemas identisch                                |
| YourSchema      | Bibliothek mit Deklarationen, die anhand des Eingabeschemas generiert wurden. Wird wie die Schema-Datei bzw. die DTD benannt |
| YourSchema Test | Testapplikationsgerüst (nur XMLSpy)                                                                                          |

#### DOM-Implementierung

Beim generierten C#-Code wird als zugrunde liegende DOM-Implementierung die .NET Standard System.XML-Bibliothek verwendet.

#### Datentypen

Das Standardmapping von XML-Schematypen auf C#-Datentypen ist:

| XML-Schema                                                          | C#                        | Anmerkungen                                                                                                    |
|---------------------------------------------------------------------|---------------------------|----------------------------------------------------------------------------------------------------------------|
| xs:string                                                           | string                    |                                                                                                                |
| xs:boolean                                                          | bool                      |                                                                                                                |
| xs:decimal                                                          | decimal                   | xs:decimal hat keine Bereichsbeschränkung und Genauigkeit, wird aus Gründen der Effizienz auf decimal gemappt. |
| xs:float, xs:double                                                 | double                    |                                                                                                                |
| xs:long                                                             | long                      |                                                                                                                |
| xs:unsignedLong                                                     | ulong                     |                                                                                                                |
| xs:int                                                              | int                       |                                                                                                                |
| xs:unsignedInt                                                      | uint                      |                                                                                                                |
| xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth | Altova.Types.Date<br>Time |                                                                                                                |
| xs:duration                                                         | Altova.Types.Dura<br>tion |                                                                                                                |
| xs:hexBinary and<br>xs:base64Binary                                 | byte[]                    | Die Kodierung und Dekodierung von Binärdateien erfolgt automatisch.                                            |
| xs:anySimpleType                                                    | string                    |                                                                                                                |

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben C#-Typ wie der entsprechende Basistyp gemappt.

#### Generierte Dokumentklasse

Zusätzlich zu den Klassen für die Typen, die im XML-Schema deklariert wurden, wird die Dokumentklasse generiert. Sie enthält als Member alle möglichen Root-Elemente sowie zusätzlich dazu die folgenden Methoden ("Doc" steht für den Namen der generierten Dokumentklasse selbst):

| Methode                                                                               | Verwendungszweck                                                                                                                                    |
|---------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| static <i>Doc</i> <b>LoadFromFile</b> (string fileName)                               | Lädt ein XML-Dokument aus einer Datei                                                                                                               |
| static <i>Doc</i> <b>LoadFromString</b> (string xml)                                  | Lädt ein XML-Dokument aus einem String                                                                                                              |
| static <i>Doc</i> <b>LoadFromBinary</b> (byte[] xml)                                  | Lädt ein XML-Dokument aus einem Byte Array                                                                                                          |
| void <b>SaveToFile</b> (string fileName, bool prettyPrint, string encoding = "UTF-8") | Speichert ein XML-Dokument in einer Datei mit der definierten Kodierung. Ist keine Kodierung definiert, wird UTF-8 verwendet.                       |
| string <b>SaveToString</b> (bool prettyPrint)                                         | Speichert ein XML-Dokument in einem String                                                                                                          |
| byte[] <b>SaveToBinary</b> (bool prettyPrint, string encoding = "UTF-8")              | Speichert ein XML-Dokument in einem Byte Array mit der angegebenen Kodierung. Wird keine Kodierung angegeben, wird UTF-8 verwendet                  |
| static <i>Doc</i> <b>CreateDocument</b> ()                                            | Erstellt ein neues, leeres XML-Dokument                                                                                                             |
| static <i>Doc</i> <b>CreateDocument</b> (string encoding)                             | Erstellt ein neues, leeres XML-Dokument mit Kodierung des Typs "encoding".                                                                          |
| void <b>SetSchemaLocation</b> (string schemaLocation)                                 | Fügt ein xsi:schemaLocation oder ein xsi:noNamespaceSchemaLocation Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein |
| void <b>SetDTDLocation</b> (string dtdLocation)                                       | Fügt eine DOCTYPE-Deklaration mit der definierten System-ID hinzu. Es muss bereits ein Root-Element vorhanden sein.                                 |

### Generierte Typklasse

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für simple und mixed Types werden eine **Value** Eigenschaft generiert, um den Textinhalt abzurufen. Für simple types mit Enumeration Facets können **getEnumerationValue()** und **setEnumerationValue(int)** zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des weiteren können Sie mit Hilfe der Eigenschaft **StaticInfo()** Schemainformationen als **Altova.Xml.Meta.SimpleType** oder **Altova.Xml.Meta.ComplexType** (siehe unten) abrufen.

Durch die XML Schema-Ableitung von complexTypes nach Extension werden in der generierten Bibliothek abgeleitete Klassen erzeugt.

### Generierte Member Attribut-Klasse

Für jedes Member Attribut eines Typs wird eine Klasse mit den folgenden Methoden oder Eigenschaften erstellt.

| Methode/Eigenschaft                | Verwendungszweck                                                                                                                                                                                                                                                        |
|------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| boolean <b>exists</b> ()           | Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist                                                                                                                                                                                                            |
| void <b>remove</b> ()              | Entfernt das Attribut aus dem übergeordneten Element                                                                                                                                                                                                                    |
| <i>AttributeType</i> <b>Value</b>  | Definiert den Attributwert oder ruft ihn ab.                                                                                                                                                                                                                            |
| <i>int</i> <b>EnumerationValue</b> | Wird nur für Enumeration Types generiert. Ruft den Attributwert ab oder definiert ihn und zwar anhand einer der Konstanten, die für die möglichen Werte generiert wurden. Gibt <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |

|                                       |                                                                           |
|---------------------------------------|---------------------------------------------------------------------------|
| Altova.Xml.Meta.Attribute <b>Info</b> | Gibt ein Objekt zum Abfragen der Schemainformationen zurück (siehe unten) |
|---------------------------------------|---------------------------------------------------------------------------|

### Generierte Member Element-Klasse

Für jedes Member Element eines Typs wird eine Klasse mit den folgenden Methoden oder Eigenschaften erstellt. Die Klasse implementiert die Standard System.Collections.IEnumerable Schnittstelle, damit sie mit der *foreach* Anweisung verwendet werden kann.

| Methode/Eigenschaft                                    | Verwendungszweck                                                                                                                                                                                                                                                       |
|--------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>MemberType</i> <b>First</b>                         | Gibt die erste Instanz des Member Elements zurück                                                                                                                                                                                                                      |
| <i>MemberType</i> <b>this</b> [int index]              | Gibt das vom Index definierte Member Element zurück                                                                                                                                                                                                                    |
| <i>MemberType</i> <b>At</b> (int index)                | Gibt das vom Index definierte Member Element zurück                                                                                                                                                                                                                    |
| <i>MemberType</i> <b>Last</b>                          | Gibt die letzte Instanz des Member Elements zurück                                                                                                                                                                                                                     |
| <i>MemberType</i> <b>Append</b> ()                     | Erstellt ein neues Element und hängt es an das übergeordnete Element an                                                                                                                                                                                                |
| bool <b>Exists</b>                                     | Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist                                                                                                                                                                                                 |
| int <b>Count</b>                                       | Gibt die Anzahl der Elemente zurück                                                                                                                                                                                                                                    |
| void <b>Remove</b> ()                                  | Löscht alle Instanzen des Elements aus dem übergeordneten Element                                                                                                                                                                                                      |
| void <b>RemoveAt</b> (int index)                       | Löscht die Instanz des durch den Index definierten Elements                                                                                                                                                                                                            |
| System.Collections.IEnumerator <b>GetEnumerator</b> () | Gibt ein Objekt für iterierende Instanzen des Member Elements zurück                                                                                                                                                                                                   |
| <i>MemberType</i> <b>Value</b>                         | Definiert den Elementinhalt oder ruft ihn ab. (wird nur generiert, wenn das Element mixed oder simple Content haben kann)                                                                                                                                              |
| int <b>EnumerationValue</b>                            | Wird nur für Enumeration Types generiert. Ruft den Elementwert ab oder definiert ihn und zwar anhand einer der Konstanten, die für die möglichen Werte generiert wurden. Gibt <b>Invalid</b> , wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt. |
| Altova.Xml.Meta.Element <b>Info</b>                    | Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe unten)                                                                                                                                                                                              |

### Erstellen von XML-Dokumenten

Mit der generierten Codebibliothek wird die Erstellung von XML-Dateien sehr einfach. Beginnen wir mit einem XML-Beispieldatei für das [Beispielschema](#):

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" xmlns="
http://www.nanonull.com/LibrarySample" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
 <Book ISBN="0764549642" Variant="Paperback">
 <Title>The XML Spy Handbook</Title>
 <Author>Altova</Author>
 </Book>
</Library>
```

Zur Erstellung dieser Datei wurde der folgende C#-Code verwendet. Die Klassen Library2, LibraryType und BookType werden anhand des [Schemas](#) generiert und stellen das gesamte Dokument, den Typ des <Library> Elements und den complexType BookType dar, der der Typ des <Book> Elements ist. Beachten Sie, dass die Funktion zur automatischen Vermeidung von Namenskonflikten die Bibliotheksklasse in Library2 umbenannt hat, um eventuelle Konflikte mit dem Namespace-Namen der Bibliothek zu vermeiden.

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" xmlns="
http://www.nanonull.com/LibrarySample" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
 <Book ISBN="0764549642" Variant="Paperback">
 <Title>The XML Spy Handbook</Title>
 <Author>Altova</Author>
 </Book>
</Library>
```

Beachten Sie, dass alle Elemente durch Aufruf von append() erstellt werden und dass Attribute (wie in diesem Beispiel ISBN) ganz einfach wie Struktur-Member aufgerufen werden können.

### Lesen von XML-Dokumenten

Im folgenden Beispiel wird die soeben erstellte Datei gelesen. Es werden die verschiedenen Methoden zum Lesen von XML-Dateien gezeigt:

```
using Library;
...
protected static void Example()
{
 // load XML document
 Library2 libDoc = Library2.LoadFromFile("Library1.xml");

 // get the first (and only) root element <Library>
 LibraryType lib = libDoc.Library3.First;

 // it is possible to check whether an element exists:
 if (!lib.Book.Exists)
 {
 Console.WriteLine("This library is empty.");
 return;
 }

 // iteration: for each <Book>...
 foreach (BookType book in lib.Book)
 {
 // output values of ISBN attribute and (first and only)
 title element
 Console.WriteLine("ISBN: " + book.ISBN.Value);
 Console.WriteLine("Title: " + book.Title.First.Value);

 // for each <Author>...
 foreach (xs.stringType author in book.Author)
 Console.WriteLine("Author: " + author.Value);

 // alternative: use count and index
 for (int j = 0; j < book.Author.Count; ++j)
 Console.WriteLine("Author: " + book.Author[j
].Value);
 }
}
```

Beachten Sie: Der Typ des book.Author Member: xs.stringType ist eine generierte Klasse für jedes Element mit einfachem String-Inhalt. Ähnliche Klassen werden für alle anderen XML-Schematypen generiert.

Sie können natürlich auch ein Element laden, es durch Hinzufügen oder Löschen von Elementen und Attributen ändern und die geänderte Datei anschließend wieder speichern.

### Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace com.altova sind die folgenden Ausnahmeereignisklassen definiert:

| Klasse                         | Basisklasse         | Beschreibung                                                                           |
|--------------------------------|---------------------|----------------------------------------------------------------------------------------|
| ConversionException            | Exception           | Ausnahmeereignis aufgrund eines Typkonvertierungsfehlers                               |
| StringParseException           | ConversionException | Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden. |
| DataSourceUnavailableException | System.Exception    | Beim Laden einer XML-Instanz ist ein Problem aufgetreten.                              |
| DataTargetUnavailableException | System.Exception    | Beim Speichern einer XML-Instanz ist ein Problem aufgetreten.                          |

Zusätzlich dazu werden die folgenden Java-Ausnahmeereignisse verwendet:

| Klasse                      | Beschreibung                                                                                                         |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------|
| System.Exception            | Basisklasse für Runtime-Fehler                                                                                       |
| System.ArgumentException    | Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen, oder es gab einen Fehler bei einer Typkonvertierung. |
| System.FormatException      | Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden.                               |
| System.InvalidCastException | Ein Wert kann nicht in einen anderen Typ umgewandelt werden.                                                         |
| System.OverflowException    | Ein Quellwert kann in der Ziel-Domain nicht dargestellt werden.                                                      |

### Aufrufen von Metadaten

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden. Alle Methoden werden als const. deklariert. Die Methoden, die eine der Metadatenklassen zurückgeben, geben NULL-Objekt zurück, wenn die entsprechende Eigenschaft nicht existiert.

#### Altova.Xml.Meta.SimpleType

| Methode/Eigenschaft            | Beschreibung                                     |
|--------------------------------|--------------------------------------------------|
| SimpleType BaseType            | Gibt den Base Type dieses Typs zurück            |
| string LocalName               | Gibt den lokalen Namen des Typs zurück           |
| string NamespaceURI            | Gibt die Namespace URI des Typs zurück           |
| XmlQualifiedName QualifiedName | Gibt den qualifizierten Namen dieses Typs zurück |
| int MinLength                  | Gibt den Wert dieses Facet zurück                |
| int MaxLength                  | Gibt den Wert dieses Facet zurück                |
| int Length                     | Gibt den Wert dieses Facet zurück                |
| int TotalDigits                | Gibt den Wert dieses Facet zurück                |
| int FractionDigits             | Gibt den Wert dieses Facet zurück                |
| string MinInclusive            | Gibt den Wert dieses Facet zurück                |

|                                  |                                                                                                                       |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| string <b>MinExclusive</b>       | Gibt den Wert dieses Facet zurück                                                                                     |
| string <b>MaxInclusive</b>       | Gibt den Wert dieses Facet zurück                                                                                     |
| string <b>MaxExclusive</b>       | Gibt den Wert dieses Facet zurück                                                                                     |
| string[] <b>Enumerations</b>     | Gibt eine Liste aller Enumeration Facets zurück                                                                       |
| string[] <b>Patterns</b>         | Gibt eine Liste aller Pattern Facets zurück                                                                           |
| WhitespaceType <b>Whitespace</b> | Gibt den Wert des Whitespace Facet zurück, der einer der folgenden ist:<br>Unknown<br>Preserve<br>Replace<br>Collapse |

**Altova.Xml.Meta.ComplexType**

| Methode/Eigenschaft                                                    | Beschreibung                                                               |
|------------------------------------------------------------------------|----------------------------------------------------------------------------|
| Attribute[] <b>Attributes</b>                                          | Gibt eine Liste aller Attribute zurück                                     |
| Element[] <b>Elements</b>                                              | Gibt eine Liste aller Elemente zurück                                      |
| ComplexType <b>BaseType</b>                                            | Gibt den Base Type dieses Typs zurück                                      |
| string <b>LocalName</b>                                                | Gibt den lokalen Namen des Typs zurück                                     |
| string <b>NamespaceURI</b>                                             | Gibt die Namespace URI des Typs zurück                                     |
| XmlQualifiedName <b>QualifiedName</b>                                  | Gibt den qualifizierten Namen dieses Typs zurück                           |
| SimpleType <b>ContentType</b>                                          | Gibt den simpleType des Inhalts zurück                                     |
| Element <b>FindElement</b> (string localName, string namespaceURI)     | Sucht das Element mit dem definierten lokalen Namen und der Namespace URI  |
| Attribute <b>FindAttribute</b> (string localName, string namespaceURI) | Sucht das Attribut mit dem definierten lokalen Namen und der Namespace URI |

**Altova.Xml.Meta.Element**

| Methode/Eigenschaft                   | Beschreibung                                                                                                                                                                                                                                             |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ComplexType <b>DataType</b>           | Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie GetContentType() des zurückgegebenen Objekts, um den simple content type abzurufen. |
| int <b>MinOccurs</b>                  | Gibt den im Schema definierten minOccurs Wert zurück                                                                                                                                                                                                     |
| int <b>MaxOccurs</b>                  | Gibt den im Schema definierten maxOccurs Wert zurück                                                                                                                                                                                                     |
| string <b>LocalName</b>               | Gibt den lokalen Namen des Elements zurück                                                                                                                                                                                                               |
| string <b>NamespaceURI</b>            | Gibt die Namespace URI des Elements zurück                                                                                                                                                                                                               |
| XmlQualifiedName <b>QualifiedName</b> | Gibt den qualifizierten Namen des Elements zurück                                                                                                                                                                                                        |

**Altova.Xml.Meta.Attribute**

| Methode/Eigenschaft                   | Beschreibung                                           |
|---------------------------------------|--------------------------------------------------------|
| SimpleType <b>DataType</b>            | Gibt den Typ des Attributinhalt zurück                 |
| bool <b>Required()</b>                | Gibt "true" zurück, wenn das Attribut erforderlich ist |
| string <b>LocalName</b>               | Gibt den lokalen Namen des Attributs zurück            |
| string <b>NamespaceURI</b>            | Gibt die Namespace URI des Attributs zurück            |
| XmlQualifiedName <b>QualifiedName</b> | Gibt den qualifizierten Namen des Attributs zurück.    |



### 31.6.5 Verwendung von generiertem Code, der mit älteren Versionen kompatibel ist

#### Kompatibilitätsmodus

Der Code Generator wurde in den letzten Altova Produktversionen immer wieder verbessert, manchmal auf eine Weise, dass der Code, der die generierten Bibliotheken verwendet, geändert werden muss. Um die Kompatibilität von bestehendem Code, der Bibliotheken verwendet, die mit früheren Releases von MapForce generiert wurden, zu gewährleisten, können Sie beim Kompatibilitätsmodus die gewünschte Release einstellen. Standardmäßig ist Version 2007r3 eingestellt, wodurch die neueste Version der in früheren Kapiteln beschriebenen Code-Bibliotheken erstellt wird.

In den folgenden Kapiteln wird beschrieben, wie die generierten Klassen bei Auswahl einer der folgenden Kompatibilitätsoptionen verwendet werden:

- Code kompatibel mit Version 2005r3
- Code kompatibel mit Version 2006 bis 2007

Bitte beachten Sie, dass diese Kompatibilitätsoptionen in zukünftigen Versionen entfernt werden.

#### Version 2005r3:

Code, der für XML-Schemas bis V2005R3 generiert wurde, verwendet eine statische DOM-Dokumentinstanz als übergeordnetes Element für alle Nodes. Auf diese Art können freistehende Nodes erstellt werden, die später zum Dokument hinzugefügt werden können. Es kann dadurch aber auch zu Problemen mit der Speicherverwaltung oder dem Multi-Threading kommen.

```
{
 LibraryType lib = new LibraryType();
 BookType book = new BookType();
 book.addISBN(new SchemaString("0764549642"));
 book.addTitle(new SchemaString("The XML Spy Handbook"));
 book.addAuthor(new SchemaString("Altova"));
 lib.addBook(book);

 LibraryDoc doc = new LibraryDoc();
 doc.setRootElementName("http://www.nanonull.com/LibrarySample",
"Library");
 doc.setSchemaLocation("Library.xsd"); // optional
 doc.save("Library1.xml", lib);
}
```

#### Version 2006 bis 2007:

In C#- und Java-Code, der mit Version 2006 bis 2007 generiert wurde, wird der Standard (Nicht-Parameter)-Konstruktor nicht verwendet, da er keine Referenz auf ein DOM-Dokument anlegt. Es wird zuerst ein neues Dokument erstellt, welches anschließend vom Root-Node aus referenziert werden kann. Alle neuen Child-Dokumente müssen anschließend ebenfalls im korrekten Dokument erstellt werden. Dazu werden die generierten Factory-Funktionen namens newXXX verwendet, wie im Beispielcode unten gezeigt.

```
{
 LibraryDoc doc = new LibraryDoc();
 doc.setSchemaLocation("Library.xsd"); // optional

 // create root element (with no namespace prefix)
 LibraryType lib = new LibraryType(doc, "
http://www.nanonull.com/LibrarySample", "", "Library");
}
```

```

 BookType book = lib.newBookType(); // factory functions are
generated for all members of a complex type
 book.addISBN(new SchemaString("0764549642"));
 book.addTitle(new SchemaString("The XML Spy Handbook"));
 book.addAuthor(new SchemaString("Altova"));
 lib.addBook(book);

 doc.save("Library1.xml", lib);
}

```

#### Version 2007r3:

Der von Version 2007r3 generierte Code wurde komplett überarbeitet, um Multithreading zu unterstützen, Namenskonflikte zu vermeiden und die Verwendung der Bibliotheken zu vereinfachen. Nähere Informationen dazu finden Sie unter [Verwendung der generierten Codebibliothek](#).

#### Erzeugen von XML-Dateien (XMLSpy 2006 - 2007)

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Generiere Code kompatibel mit V2006-2007** im Dialogfeld "Codegenerierung" beschrieben.

#### Java:

Um Ihnen eine Übersicht zu geben, finden Sie unten den Code für die Erzeugung der Testdatei:

```

protected static void example() throws Exception {
 LibraryDoc doc = new LibraryDoc();
 doc.setSchemaLocation("Library.xsd"); // optional
 LibraryType lib = new LibraryType(doc,
"http://www.nanonull.com/LibrarySample", "", "Library");
 BookType book = lib.newBook();
 book.addISBN(new SchemaString("0764549642"));
 book.addTitle(new SchemaString("The XML Spy Handbook"));
 book.addAuthor(new SchemaString("Altova"));
 lib.addBook(book);
 doc.save("Library1.xml", lib);
}

```

Den generierten Klassen liegt die folgende Idee zugrunde:

Erstellen eines Dokuments und Verknüpfung mit dem Root-Element des generierten Schemas, sodass die Baumstruktur im Dokument gespeichert ist.

Zuerst wird ein neues Dokument erstellt:

```
LibraryDoc doc = new LibraryDoc();
```

Optional kann auch der Schemapfad definiert werden.

```
doc.setSchemaLocation("Library.xsd");
```

Wir erstellen das Root-Element, übergeben das bereits generierte Dokumentobjekt, den Namespace von `library.xsd` und den Namen des Root-Elements im Schema an die Methode:

```
LibraryType lib = new LibraryType(doc, "http://www.nanonull.com/LibrarySample", "", "Library");
```

**LibraryType** ist eine Klasse, die generiert wurde. Sie ist dem Element-Namen `<xs:element name="Library">` in der Schema-Datei ähnlich. Dies ist das Root-Element der generierten XML-Dateien (wie beim Generieren einer XML-Beispieldatei).

Wir haben nun ein neues Library-Objekt. Nun müssen wir dieses nur noch mit Book-Objekten befüllen. Wenn Sie sich das Schema noch einmal ansehen, werden Sie feststellen, dass "Books" als eine Sequenz innerhalb der Library gespeichert werden.

Der nächste Schritt ist, ein Objekt von **BookType** zu erstellen und es auszufüllen.

```
BookType book = lib.newBook();
```

Definition von BookType in der Schema-Datei:

```
<xs:complexType name="BookType">
 <xs:sequence>
 <xs:element name="Title" type="xs:string"/>
 <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute name="ISBN" type="xs:string" use="required"/>
</xs:complexType>
```

Wir fügen nun eine ISBN-Nummer, einen Titel und einen Autor ein.

```
book.addISBN(new SchemaString("0764549642"));
book.addTitle(new SchemaString("The XML Spy Handbook"));
book.addAuthor(new SchemaString("Altova"));
```

Wir haben das Objekt "BookType" nun befüllt und müssen es nun zur Bibliothekssequenz hinzufügen:

```
lib.addBook(book);
```

Speichern Sie die Datei nun. Der erste Parameter ist der Dateiname, der zweite ist das Root-Element selbst.

```
doc.save("Library1.xml", lib);
```

## C++:

Die C++-Implementierung ist der Java-Implementierung sehr ähnlich. Daher werden wir nur auf die Unterschiede zwischen den beiden näher eingehen.

Die oberhalb der Hauptfunktion einzufügende Funktion lautet wie folgt:

```
void Example()
{
 CLibraryDoc doc;
 CLibraryType lib;
 doc.SetRootElementName(_T("http://www.nanonull.com/LibrarySample"), _T(
"Library"));
 doc.SetSchemaLocation(_T("Library.xsd")); // optional
 CBookType book;
 book.AddISBN(_T("0764549642"));
 book.AddTitle(_T("The XML Spy Handbook"));
 book.AddAuthor(_T("Altova"));
```

```
lib.AddBook(book);
doc.Save(_T("Library1.xml"), lib);
```

Als erstes fällt auf, dass alle Klassen mit einem großgeschriebenen C beginnen. Der Rest der Klassennamen bleibt gleich. Alle Member Function-Namen beginnen mit einem Großbuchstaben.

Seien Sie bitte vorsichtig mit den `_T`-Makros. Das Makro stellt sicher, dass die String-Konstante korrekt gespeichert wird, ob Sie nun für Unicode oder Nicht-Unicode-Programme kompilieren.

## C#:

```
protected static void Example()
{
 LibraryDoc doc = new LibraryDoc();
 doc.SetSchemaLocation("Library.xsd"); // optional
 // create root element with no namespace prefix
 LibraryType lib = new LibraryType(doc, "
http://www.nanonull.com/LibrarySample", "", "Library");
 BookType book = lib.NewBook(); // factory functions are generated for all
members of a complex type
 book.AddISBN(new SchemaString("0764549642"));
 book.AddTitle(new SchemaString("The XML Spy Handbook"));
 book.AddAuthor(new SchemaString("Altova"));
 lib.AddBook(book);
 doc.Save("Library1.xml", lib);
}
```

## Erzeugen von XML-Dateien (XMLSpy 2005)

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Mit V2005R3 kompatiblen Code generieren** aus dem Dialogfeld "Codegenerierung" beschrieben.

## Java:

Um Ihnen eine Übersicht zu geben, finden Sie unten den Code für die Erzeugung der Testdatei:

```
protected static void example() {
 LibraryType lib = new LibraryType();
 BookType book = new BookType();
 book.addISBN(new SchemaString("0764549642"));
 book.addTitle(new SchemaString("The XML Spy Handbook"));
 book.addAuthor(new SchemaString("Altova"));
 lib.addBook(book);

 LibraryDoc doc = new LibraryDoc();
 doc.setRootElementName("http://www.nanonull.com/LibrarySample", "Library"
);
 doc.setSchemaLocation("Library.xsd"); // optional
 doc.save("Library1.xml", lib);
}
```

Den generierten Klassen liegt die folgende Idee zugrunde:  
Erstellen eines Dokuments und Verknüpfung mit dem Root-Element des generierten Schemas, sodass die Baumstruktur im Dokument gespeichert ist.

Die erste Zeile:

```
LibraryType lib = new LibraryType();
```

**LibraryType** ist eine Klasse, die generiert wurde. Sie ist dem Element-Namen `<xs:element name="Library">` in der Schema-Datei ähnlich. Dies ist das Root-Element der generierten XML-Dateien (wie beim Generieren einer XML-Beispieldatei).

Wir haben nun ein neues Library-Objekt. Nun müssen wir dieses nur noch mit Book-Objekten befüllen. Wenn Sie sich das Schema noch einmal ansehen, werden Sie feststellen, dass "Books" als eine Sequenz innerhalb der Library gespeichert werden.

Der nächste Schritt ist, ein Objekt von **BookType** zu erstellen und es auszufüllen.

```
BookType book = new BookType();
```

Definition von BookType in der Schema-Datei:

```
<xs:complexType name="BookType">
 <xs:sequence>
 <xs:element name="Title" type="xs:string"/>
 <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute name="ISBN" type="xs:string" use="required"/>
</xs:complexType>
```

Wir fügen nun eine ISBN-Nummer, einen Titel und einen Autor ein.

```
book.addISBN(new SchemaString("0764549642"));
book.addTitle(new SchemaString("The XML Spy Handbook"));
book.addAuthor(new SchemaString("Altova"));
```

Wir haben das Objekt "BookType" nun befüllt und müssen es nun zur Sequenz von **internationalPriceTypes** in Test\_Empty hinzufügen:

```
lib.addBook(book);
```

Fertig, die Baumstruktur ist erstellt! Nun müssen wir nur noch ein neues Dokument erstellen und den Namespace und Namen des Root-Elements definieren:

```
LibraryDoc doc = new LibraryDoc();
doc.setRootElementName("http://www.nanonull.com/LibrarySample", "Library"
);
```

Optional können wir auch den Speicherort des Schemas festlegen.

```
doc.setSchemaLocation("Library.xsd");
```

Speichern Sie die Datei nun. Der erste Parameter ist der Dateiname, der zweite ist das Root-Element selbst.

```
doc.save("Library1.xml", lib);
```

**C++:**

Die C++-Implementierung ist der Java-Implementierung sehr ähnlich. Daher werden wir nur auf die Unterschiede zwischen den beiden näher eingehen.

Die oberhalb der Hauptfunktion einzufügende Funktion lautet wie folgt:

```
void Example()
{
 CLibraryType lib;
 CBookType book;
 book.AddISBN(_T("0764549642"));
 book.AddTitle(_T("The XML Spy Handbook"));
 book.AddAuthor(_T("Altova"));
 lib.AddBook(book);

 CLibraryDoc doc;
 doc.SetRootElementName(_T("http://www.nanonull.com/LibrarySample"), _T(
"Library"));
 doc.SetSchemaLocation(_T("Library.xsd")); // optional
 doc.Save(_T("Library1.xml"), lib);
}
```

Als erstes fällt auf, dass alle Klassen mit einem großgeschriebenen C beginnen. Der Rest der Klassennamen bleibt gleich. Alle Member Function-Namen beginnen mit einem Großbuchstaben.

Seien Sie bitte vorsichtig mit den `_T`-Makros. Das Makro stellt sicher, dass die String-Konstante korrekt gespeichert wird, ob Sie nun für Unicode oder Nicht-Unicode-Programme kompilieren.

## C#:

```
protected static void Example()
{
 LibraryType lib = new LibraryType();
 BookType book = new BookType();
 book.AddISBN(new SchemaString("0764549642"));
 book.AddTitle(new SchemaString("The XML Spy Handbook"));
 book.AddAuthor(new SchemaString("Altova"));
 lib.AddBook(book);

 LibraryDoc doc = new LibraryDoc();
 doc.SetRootElementName("http://www.nanonull.com/LibrarySample", "Library"
);
 doc.SetSchemaLocation("Library.xsd"); // optional
 doc.Save("Library1.xml", lib);
}
```

## Öffnen und Parsen vorhandener XML-Dateien (XMLSpy 2006 - 2007)

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Mit V2006-2007 kompatiblen Code generieren** aus dem Dialogfeld "Codegenerierung" beschrieben.

Wir werden nun mit der mit unserem Beispielpogramm erstellten Datei **Library1.xml** arbeiten. Stellen Sie sicher, dass diese Datei vorhanden ist oder ändern Sie im load-Befehl den Pfad zur Datei.

**Java:**

Beispiel-Quellcode:

```

protected static void example2() {

 LibraryDoc doc = new LibraryDoc();
 LibraryType lib = new LibraryType(doc.load("Library1.xml"));

 if (!lib.hasBook()) {
 System.out.println("This library is empty");
 return;
 }

 for (int i = 0; i < lib.getBookCount(); i++) {
 BookType book;
 try {
 book = lib.getBookAt(i);

 System.out.println("ISBN: " + book.getISBN().toString());
 System.out.println("Title: " + book.getTitle().toString());

 for (int j = 0; j < book.getAuthorCount(); j++) {
 System.out.println("Author: " +
book.getAuthorAt(j).toString());
 }
 } catch (Exception e) {
 e.printStackTrace();
 }
 }
}

```

Fügen Sie in der **main()** function einen Call zu:

```

 example2();
 hinzu nach
 example();

```

Der Rest ist relativ einfach.

Erstellen Sie zuerst eine neue Variable vom Typ Document sowie eine für das Root-Element, das sofort durch die Funktion **doc.load** befüllt wird. Der einzige dazu benötigte Parameter ist der Pfad zur verwendeten XML-Datei.

```

LibraryDoc doc = new LibraryDoc();
LibraryType lib = new LibraryType(doc.load("Library1.xml"));

```

Wir haben nun ein Dokument und wissen, dass "lib" eine Sequenz von BookType-Objekten enthält.

Der nächste Schritt ist, die Werte aller Book-Elemente auszugeben:

```

if(!lib.hasBook()) {

```

Warum soll auf Objekte überprüft werden? Der Call zeigt nur, dass eine Funktion existiert, die dies für Sie überprüft.

Iterieren Sie durch die Books und geben Sie deren Werte aus:

```

 for(int i = 0; i < lib.getBookCount(); i++) {
 BookType book = lib.getBookAt(i);

 System.out.println("ISBN: " + book.getISBN().asString());
 System.out.println("Title: " + book.getTitle().asString());

 for(int j = 0; j < book.getAuthorCount(); j++) {
 System.out.println("Author: " + book.getAuthorAt(j).asString()
);
 }
 }
}

```

Wir sind nun fertig. Alle Werte werden in Ihrem Ausgabefenster aufgelistet.

### C++:

In diesem Abschnitt werden nur die Unterschiede zum Java-Code erläutert.

```

void Example2()
{
 CLibraryDoc doc;
 CLibraryType lib = doc.Load(_T("Library1.xml"));

 if (!lib.HasBook())
 {
 cout << "This library is empty" << endl;
 return;
 }

 for (int i = 0; i < lib.GetBookCount(); i++)
 {
 CBookType book = lib.GetBookAt(i);
 cout << "ISBN: " << book.GetISBN() << endl;
 cout << "Title: " << book.GetTitle() << endl;

 for (int j = 0; j < book.GetAuthorCount(); j++)
 {
 cout << "Author: " << book.GetAuthorAt(j) << endl;
 }
 }
}

```

Alle Member Functions beginnen mit einem Großbuchstaben und alle Klassennamen beginnen mit einem C.

### C#:

In diesem Abschnitt werden nur die Unterschiede zum Java-Code erläutert.

```

protected static void Example2()
{
 LibraryDoc doc = new LibraryDoc();
 LibraryType lib = new LibraryType(doc.Load("Library1.xml"));

 if (!lib.HasBook())
 {
 Console.WriteLine("This library is empty");
 }
}

```



```

 return;
 }

 for (int i = 0; i < lib.GetBookCount(); i++)
 {
 BookType book = lib.GetBookAt(i);
 Console.WriteLine("ISBN: {0}", book.GetISBN());
 Console.WriteLine("Title: {0}", book.GetTitle());

 for(int j = 0; j < book.GetAuthorCount(); j++)
 {
 Console.WriteLine("Author: {0}", book.GetAuthorAt(j));
 }
 }
}

```

Alle Member Functions beginnen mit einem Großbuchstaben.

### Öffnen und Parsen vorhandener XML-Dateien (XMLSpy 2005)

**Bitte beachten Sie:** In diesem Kapitel wird die Codegenerierung bei Verwendung der Option **Mit V2005R3 kompatiblen Code generieren** aus dem Dialogfeld "Codegenerierung" beschrieben.

Wir werden nun mit der mit unserem Beispielprogramm erstellten Datei **Library1.xml** arbeiten. Stellen Sie sicher, dass diese Datei vorhanden ist oder ändern Sie im load-Befehl den Pfad zur Datei.

### Java:

Beispiel-Quellcode:

```

protected static void example2() {
 LibraryDoc doc = new LibraryDoc();
 LibraryType lib = new LibraryType(doc.load("Library1.xml"));

 if(!lib.hasBook()) {
 System.out.println("This library is empty");
 return;
 }

 for(int i = 0; i < lib.getBookCount(); i++) {
 BookType book = lib.getBookAt(i);
 System.out.println("ISBN: " + book.getISBN().asString());
 System.out.println("Title: " + book.getTitle().asString());

 for(int j = 0; j < book.getAuthorCount(); j++) {
 System.out.println("Author: " + book.getAuthorAt(j
).asString());
 }
 }
}

```

Fügen Sie in der **main()** function einen Call zu:

```

 example2();
 hinzu nach

```

```
example1();
```

Der Rest ist relativ einfach.

Erstellen Sie zuerst eine neue Variable vom Typ Document sowie eine für das Root-Element, das sofort durch die Funktion **doc.load** befüllt wird. Der einzige dazu benötigte Parameter ist der Pfad zur verwendeten XML-Datei.

```
LibraryDoc doc = new LibraryDoc();
LibraryType lib = new LibraryType(doc.load("Library1.xml"));
```

Wir haben nun ein Dokument und wissen, dass "lib" eine Sequenz von BookType-Objekten enthält.

Der nächste Schritt ist, die Werte aller Book-Elemente auszugeben:

```
if(!lib.hasBook()) {
```

Warum soll auf Objekte überprüft werden? Der Call zeigt nur, dass eine Funktion existiert, die dies für Sie überprüft.

Iterieren Sie durch die Books und geben Sie deren Werte aus:

```
for(int i = 0; i < lib.getBookCount(); i++) {
 BookType book = lib.getBookAt(i);

 System.out.println("ISBN: " + book.getISBN().asString());
 System.out.println("Title: " + book.getTitle().asString());

 for(int j = 0; j < book.getAuthorCount(); j++) {
 System.out.println("Author: " + book.getAuthorAt(j).asString());
 }
};
```

Wir sind nun fertig. Alle Werte werden in Ihrem Ausgabefenster aufgelistet.

## C++:

In diesem Abschnitt werden nur die Unterschiede zum Java-Code erläutert.

```
void Example2()
{
 CLibraryDoc doc;
 CLibraryType lib = doc.Load(_T("Library1.xml"));

 if(!lib.HasBook())
 {
 cout << "This library is empty" << endl;
 return;
 }

 for(int i = 0; i < lib.GetBookCount(); i++)
 {
 CBookType book = lib.GetBookAt(i);

 cout << "ISBN: " << book.GetISBN() << endl;
 cout << "Title: " << book.GetTitle() << endl;
 }
}
```

```

 for(int j = 0; j < book.GetAuthorCount(); j++)
 {
 cout << "Author: " << book.GetAuthorAt(j) << endl;
 }
 }
}

```

Alle Member Functions beginnen mit einem Großbuchstaben und alle Klassennamen beginnen mit einem C.

## C#:

In diesem Abschnitt werden nur die Unterschiede zum Java-Code erläutert.

```

protected static void Example2()
{
 LibraryDoc doc = new LibraryDoc();
 LibraryType lib = new LibraryType(doc.Load("Library1.xml"));

 if(!lib.HasBook())
 {
 Console.WriteLine("This library is empty");
 return;
 }

 for(int i = 0; i < lib.GetBookCount(); i++)
 {
 BookType book = lib.GetBookAt(i);

 Console.WriteLine("ISBN: {0}", book.GetISBN());
 Console.WriteLine("Title: {0}", book.GetTitle());

 for(int j = 0; j < book.GetAuthorCount(); j++)
 {
 Console.WriteLine("Author: {0}", book.GetAuthorAt(j));
 }
 }
}

```

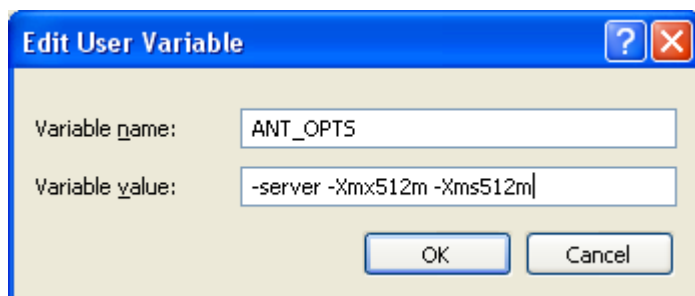
Alle Member Functions beginnen mit einem Großbuchstaben.

## 31.7 Codegenerierung Tipps

### Ausnahmeereignisse bei zu wenig Arbeitsspeicher bei der Java-Kompilierung und Lösungsmöglichkeiten

Es kommt vor, dass bei Mappings eine so große Menge von Code generiert wird, dass es bei der Kompilierung mit Ant zu einem `java.lang.OutOfMemory`-Ausnahmeereignis kommt. Um dieses Problem zu beheben:

- Fügen Sie die Umgebungsvariable **Ant\_OPTS** hinzu. Damit werden bestimmte Ant-Optionen festgelegt, wie z.B. wieviel Arbeitsspeicher dem Compiler zur Verfügung steht. Sie können Werte wie unten gezeigt hinzufügen.



- Um sicherzustellen, dass der Compiler und der generierte Code im selben Prozess wie Ant ausgeführt wird, ändern Sie das Attribut **fork** in der Datei `build.xml` in **false** um.

Je nach verfügbarem Arbeitsspeicher auf Ihrem Computer und je nach Projektgröße müssen Sie diese Werte eventuell anpassen. Nähere Informationen finden Sie in der Java VM-Dokumentation.

### Reservieren von Methodennamen:

Beim Anpassen der Codegenerierung mittels der vorgegebenen SPL-Dateien ist es unter Umständen nötig, Methodennamen zu reservieren. Gehen Sie dazu folgendermaßen vor:

- Navigieren Sie zum Unterverzeichnis zu den entsprechenden Programmiersprachen des **spl**-Unterverzeichnisses des Programminstallationsverzeichnisses z.B. `c:\Program Files\Altova\MapForce2012\spl\java`.
- Öffnen Sie entweder die Datei **settings.spl** und fügen Sie in den Abschnitt "reserve" eine neue Zeile ein, z.B. `reserve "myReservedWord"`
- Generieren Sie den Programmcode erneut.

### XML Schema-Unterstützung:

Die folgenden XML-Schema-Konstrukte werden in Code übersetzt:

- XML Namespaces: Werden in der Zielsprache auf Namespaces gemappt

#### Simple types

- Built-in XML Schema Typen: Werden auf Klassen in der Altova Typbibliothek gemappt
- Durch Extension abgeleitete
- Durch Restriction abgeleitete
- Facets
- Enumerationen
- Patterns

#### Complex types

- Built-in anyType Node
- Benutzerdefinierte complex types
- Durch Extension abgeleitete: Auf abgeleitete Klassen gemappt
- Durch Restriction abgeleitete
- Complex Inhalt
- Simple Inhalt
- Mixed Inhalt

Die folgenden komplexen XML-Schema-Funktionen **werden** in MapForce beim Generieren optionaler **Wrapper**-Klassen nicht oder nicht vollständig unterstützt:

- \* Wildcards: xs:any und xs:anyAttribute
- Inhaltsmodelle (sequence, choice, all): Kompositor der obersten Ebene ist in SPL vorhanden, wird aber von den generierten Klassen nicht umgesetzt
- Default und Fixed Values für Attribute: In SPL verfügbar, wird aber von den generierten Klassen nicht gesetzt oder umgesetzt
- Attribute xsi:type, abstract types: SetXsiType Methoden werden generiert und müssen vom Benutzer aufgerufen werden
- Union Types: Es werden nicht alle Kombinationen unterstützt
- Substitution Groups: Teilweise Unterstützung (wird wie "choice" aufgelöst)
- attribute nillable="true" und xsi:nil
- Uniqueness Constraints
- key und keyref

## 31.8 Code Generator-Optionen

Über die Menüoption **Extras | Optionen** können Sie allgemeine sowie spezifische MapForce Einstellungen konfigurieren.

"Code-Generierung":

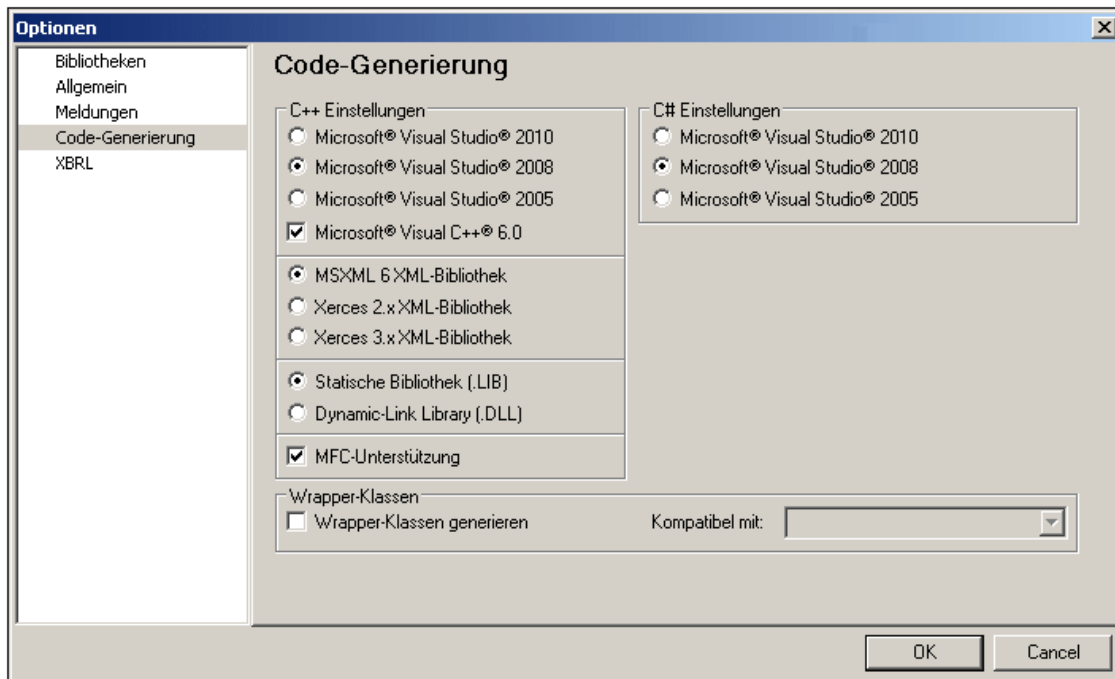
### C++-Einstellungen

- Microsoft Visual Studio 2010, 2008, 2005, Visual C++ 6.0 Projektdatei .
- Generierung von Code mit Hilfe von entweder MSXML 6.0 oder der Apache Xerces XML-Bibliothek)
- Generierung von Static Link Libraries oder Dynamic Link Libraries (DLLs)
- Generierung von Code mit oder ohne MFC-Unterstützung

### C#-Einstellungen

Wählen Sie die Art der zu generierenden Projektdatei aus:

- Microsoft Visual Studio 2010, 2008, 2005 Projektdatei
- Mono Makefile wird ab MapForce Version 2011 nicht mehr unterstützt. Benutzer von Mono können mit xbuild von Mono Visual Studio Solution-Dateien verwenden.



### Java-Einstellungen

Von MapForce generierte Webservice Implementierungen unterstützen je nach dem in der WSDL-Datei ausgewählten Binding SOAP 1.1 oder SOAP 1.2. Apache Axis2 unterstützt sowohl SOAP 1.1 als auch SOAP 1.2.

### Wrapper-Klassen:

Ab der MapForce Version 2007r3 werden vollkommen neue Code Generator-Vorlagen verwendet, für die für das Quell- und Zielschema, bzw. die Datenbankstrukturen, EDI- und Textstrukturen keine Wrapper-Klassen mehr generiert werden müssen. Dies hat den Vorteil von weniger Code, der schneller kompiliert wird, weniger Arbeitsspeicher benötigt und sich schneller ausführen lässt. Wenn Sie das generierte MapForce Mapping jedoch in Ihr eigenes Programm integrieren möchten und die XML-Quell- und/oder Zielinstanz in Ihrem Programm über eine Wrapper-Klasse aufrufen möchten, wie sie auch von XMLSpy generiert werden,

können Sie die Generierung dieser Klassen hier immer noch aktivieren.

**Kompatibilitätsmodus:**

Der Code Generator wurde in den letzten Altova Produktversionen immer wieder verbessert, manchmal auf eine Weise, dass der Code, der die generierten Bibliotheken verwendet, geändert werden muss. Um die Kompatibilität von bestehendem Code, der Bibliotheken verwendet, die mit früheren Releases von MapForce generiert wurden, zu gewährleisten, können Sie beim Kompatibilitätsmodus die gewünschte Release einstellen. Standardmäßig ist Version 2007r3 eingestellt, wodurch die neueste Version der in früheren Kapiteln beschriebenen Code-Bibliotheken erstellt wird.

- Bitte verwenden Sie nach Möglichkeit die neuen Codegenerierungsfunktionen, aktivieren Sie also die neueste Version.
- Der Kompatibilitätsmodus kann nur temporär eingestellt werden und wird in zukünftigen Versionen nicht mehr zur Verfügung stehen.

## 31.9 Der Weg zu SPL (Spy Programming Language)

In diesem Abschnitt erhalten Sie eine Übersicht über Spy Programming Language (SPL), die Template-Sprache von Code Generator.

Es wird vorausgesetzt, dass Sie bereits über Programmiererfahrung verfügen und mit Operatoren, Funktionen, Variablen und Klassen sowie den Grundzügen von in SPL häufig verwendeter objektorientierter Programmierung vertraut sind.

Im Ordner ...MapForce2012\spl finden Sie die von MapForce verwendeten Vorlagen. Anhand dieser Dateien können Sie Ihre eigenen Vorlagen entwickeln.

### Funktionsweise von Code Generator:

Die Basis von Code Generator bilden die Vorlagendateien (.spl) und das von MapForce bereitgestellte Objektmodell. Die Vorlagendateien enthalten SPL-Anweisungen zum Erstellen von Dateien, Lesen von Informationen aus dem Objektmodell und Ausführen von Berechnungen, die in der Zielprogrammiersprache auch Literalcodefragmente enthalten.

Die Vorlagendatei wird von Code Generator interpretiert und je nach Vorlage wird eine **.cpp, .c, .java, .cs** Quellcodedatei Projektdatei oder eine andere Dateiart ausgegeben. Der Quellcode wird anschließend in eine exe-Datei kompiliert. Nach Starten der exe-Datei greift diese auf die durch die Schema-Datei beschriebenen XML-Daten zu.

SPL-Dateien haben Zugriff auf die verschiedensten Informationen, die aus den Quellschemas geholt werden. Beachten Sie bitte, dass eine SPL-Datei nicht an ein bestimmtes Schema gebunden ist, sondern Zugriff auf alle Schemas erlaubt! Stellen Sie sicher, dass Sie Ihre SPL-Dateien generisch anlegen, Strukturen usw. vermeiden, die nur auf bestimmte Schemas zutreffen!

### Beispiel: Erstellen einer neuen Datei in SPL:

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

Dies ist ein sehr einfacher SPL-Code. Damit wird eine Datei names **test.cpp** erstellt. Das Include Statement wird innerhalb davon positioniert. Der Befehl "close" schließt und beendet die Vorlage.



### 31.9.1 Grundlegende SPL-Struktur

Eine SPL-Datei enthält Literaltext, der ausgegeben werden soll, der zwischendurch Code Generator-Anweisungen enthält.

Code Generator-Anweisungen sind in eckige Klammern eingeschlossen '[' und ']'. Innerhalb einer Klammer können mehrere Anweisungen stehen. Zusätzliche Anweisungen müssen durch eine neue Zeile oder einen Doppelpunkt ':' getrennt werden.

Gültige Beispiele sind:

```
[$x = 42
$x = $x + 1]
```

or

```
[$x = 42: $x = $x + 1]
```

#### Hinzufügen von Text zu Dateien

Text, der nicht innerhalb von '[' und ']' steht, wird direkt in die Ausgabedatei geschrieben. Falls es keine aktuelle Ausgabedatei gibt, wird der Text ignoriert (eine Anleitung zum Erstellen einer Ausgabedatei finden Sie unter [Verwendung von Dateien](#)).

Um eckige Klammern als Literale auszugeben, setzen Sie davor als Escape-Zeichen einen umgekehrten Schrägstrich: '\[' und '\]'. Um einen umgekehrten Schrägstrich auszugeben, verwenden Sie '\\.

#### Kommentare

Kommentare innerhalb eines Anweisungsblocks beginnen immer mit einem ' ' Zeichen und enden an der nächsten Zeile oder an einem Zeichen, zum Beenden eines Blocks ']'.

### 31.9.2 Deklarationen

Die folgenden Anweisungen werden beim Parsen der SPL-Vorlagendatei ausgewertet. Flusssteuerungsanweisungen wie z.B. Bedingungen, Schleifen oder Subroutinen haben **keinen** Einfluss darauf und werden immer nur einmal ausgewertet.

Die Groß- und Kleinschreibung spielt wie bei allen Schlüsselwörtern in SPL keine Rolle.

Alle diese Deklarationen müssen unbedingt innerhalb von eckigen Klammern stehen.

**map** *mapname* **key to value** [, **key to value** ]...

Diese Anweisung fügt Informationen zu einer Zuordnung hinzu. Beispiele für konkrete Anwendungen siehe unten.

**map schemanativetype** *schematype to typespec*

Der angegebene built-in XML-Schematyp wird unter Verwendung der angegebenen Formatierung auf den angegebenen Native Type oder die Klasse gemappt. Diese Einstellung gilt nur für die Codegenerierung für Version 2007r3 und höher. Typespec ist ein Native Type oder ein Klassenname, gefolgt von einem Komma, gefolgt vom der Klasseninstanz des Formatters..

Beispiel:

```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

**map type** *schematype to classname*

Der angegebene vordefinierte XML-Schematyp wird auf die angegebene Klasse gemappt. Diese Einstellung gilt nur für die Codegenerierung für Version 2007 oder niedriger.

Beispiel:

```
map type "float" to "CSchemaFloat"
```

**default setting is value**

Mit dieser Anweisung können Sie festlegen, wie Klassen- und Member-Namen vom XML-Schema abgeleitet werden.

Beachten Sie, dass die die Groß- und Kleinschreibung bei Einstellungsnamen relevant ist.

Beispiel:

```
default "InvalidCharReplacement" is "_"
```

#### Einstellungsname Erklärung

e

|                        |                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------|
| ValidFirstCharSet      | Zulässige Zeichen zum Starten eines Identifizier                                                      |
| ValidCharSet           | Zulässige Zeichen für andere Zeichen in einem Identifizier                                            |
| InvalidCharReplacement | Das Zeichen, durch das alle Zeichen in Namen ersetzt werden, die nicht im ValidCharSet enthalten sind |
| AnonTypePrefix         | Präfix für Namen von anonymen Typen*                                                                  |
| AnonTypeSuffix         | Suffix für Namen von anonymen Typen*                                                                  |
| ClassNamePrefix        | Präfix für generierte Klassennamen                                                                    |
| ClassNameSuffix        | Suffix für generierte Klassennamen                                                                    |
| EnumerationPrefix      | Präfix für Symbolkonstanten, die für Enumerationswerte deklariert wurden                              |
| EnumerationUpperCase   | "ein" für das Konvertieren der Namen der Enumerationskonstanten in Großbuchstaben                     |
| FallbackName           | Verwende diesen Namen, wenn ein Name nur aus Zeichen besteht, die sich nicht im ValidCharSet befinden |

\* Namen anonymer Typen werden aus AnonTypePrefix + Elementname + AnonTypeSuffix erzeugt.

**reserve** *word*

Fügt das angegebene Wort zur Liste der reservierte Wörter hinzu. Damit stellen Sie sicher, dass es nie als Klasse oder Member-Name generiert wird.

Beispiel:  
reserve "while"

**include** *filename*

Beispiel:  
include "Module.cpp"

Inkludiert die angegebene Datei als SPL-Quelldatei. Auf diese Art können Sie Ihre Vorlage zur einfacheren Bearbeitung und Verwaltung in mehrere Dateien aufteilen.

### 31.9.3 Variablen

Jede etwas komplexere SPL-Datei enthält Variablen. Einige Variablen sind vom Code Generator [vordefiniert](#). Sie können jederzeit neue Variablen erstellen, indem Sie diesen Werte zuweisen.

Das **\$**-Zeichen wird zur **Deklaration** oder **Verwendung** einer Variable verwendet. Vor der Variable muss immer ein **\$** stehen.

Bei Variablennamen muss die **Groß- und Kleinschreibung beachtet werden**.

Variablentypen:

- Integer - wird auch als Boolescher Wert verwendet, wobei 0 false ist und alle anderen Werte true.
- String
- Objekt - wird von MapForce bereitgestellt
- Iterator (siehe [foreach](#)-Anweisung)

Variablentypen werden durch die erste Zuweisung deklariert.

```
[$x = 0]
```

x ist nun ein Integer.

```
[$x = "teststring"]
```

x wird nun als String behandelt.

#### Strings:

String-Konstanten müssen wie im oben gezeigten Beispiel immer innerhalb von doppelten Anführungszeichen stehen. **\n** und **\t** innerhalb von doppelten Anführungszeichen werden als neue Zeile und Tabulator interpretiert, **\** ist ein literales doppeltes Anführungszeichen und **\\** ist ein umgekehrter Schrägstrich. String-Konstanten können auch mehrere Zeilen einnehmen.

Bei der Verkettung von Strings wird das **&** Zeichen verwendet.

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

#### Objekte

Objekte stehen für die Informationen, die im XML-Schema/in den XML-Schemas, den Datenbankstrukturen, Textdateien und Mappings enthalten sind. Objekte haben **Eigenschaften**, die über den **.** Operator aufgerufen werden können. In SPL können keine neuen Objekte erstellt werden (sie werden durch den Code Generator vordefiniert, aus dem Eingabe-Mapping abgeleitet). Sie können Objekte jedoch Variablen zuweisen.

Beispiel:

```
class [= $class.Name]
```

In diesem Beispiel wird das Wort "class" ausgegeben, gefolgt von einem Leerzeichen und dem Wert der Eigenschaft **Name** des Objekts **\$class**.

### 31.9.4 Vordefinierte Variablen

Nachdem eine Schema-Datei von Code Generator analysiert wurde, sind die in der Tabelle aufgelisteten Objekte im Template-Prozessor vorhanden.

Aktuelles Objektmodell (für Codekompatibilität mit Version 2007r3 und höher)

| Name         | Typ                     | Beschreibung                                                         |
|--------------|-------------------------|----------------------------------------------------------------------|
| \$schematype | integer                 | 1 für DTD, 2 für XML-Schema                                          |
| \$TheLibrary | <a href="#">Library</a> | Die anhand des XML-Schemas oder der DTD abgeleitete Bibliothek       |
| \$module     | string                  | Name des Quellschemas ohne Erweiterung                               |
| \$outputpath | string                  | Der vom Benutzer definierte Ausgabepfad oder der Standardausgabepfad |

Altes Objektmodell (für Codekompatibilität bis Version 2007)

| Name         | Typ                                     | Beschreibung                                                         |
|--------------|-----------------------------------------|----------------------------------------------------------------------|
| \$schematype | Integer                                 | 1 für DTD, 2 für XML Schema                                          |
| \$namespaces | <a href="#">Namespace</a><br>Collection | Collection von Namespace-Objekten                                    |
| \$classes    | <a href="#">Class</a> Collection        | Alle complexTypes, Elemente, ...in einer flachen Ansicht             |
| \$module     | String                                  | Name des Quellschemas ohne Erweiterung                               |
| \$outputpath | String                                  | Der vom Benutzer definierte Ausgabepfad oder der Standardausgabepfad |

Nur bei C++ Generierung:

| Name            | Typ     | Beschreibung                                                      |
|-----------------|---------|-------------------------------------------------------------------|
| \$domtype       | Integer | 1 bei MSXML, 2 bei Xerces                                         |
| \$XercesVersion | Integer | 2 bei Xerces 2.x, 3 bei Xerces 3.x                                |
| \$libtype       | Integer | 1 bei statischer LIB, 2 bei DLL                                   |
| \$mfc           | Boolean | True, wenn MFC Unterstützung aktiviert ist                        |
| \$vc6project    | Boolean | True, wenn Visual C++ 6.0 Projektdateien generiert werden sollen  |
| \$VS2005Project | Boolean | True, wenn Visual C++ 2005 Projektdateien generiert werden sollen |
| \$VS2008Project | Boolean | True, wenn Visual C++ 2008 Projektdateien generiert werden sollen |
| \$VS2010Project | Boolean | True, wenn Visual C++ 2010 Projektdateien generiert werden sollen |

Nur bei C# Generierung:

| Name                  | Typ     | Beschreibung                                                         |
|-----------------------|---------|----------------------------------------------------------------------|
| \$CreateVS2005Project | Boolean | True, wenn Visual Studio 2005 Projektdateien generiert werden sollen |
| \$CreateVS2008Project | Boolean | True, wenn Visual Studio 2008 Projektdateien generiert werden sollen |
| \$CreateVS2010Project | Boolean | True, wenn Visual Studio 2010 Projektdateien generiert werden sollen |

### 31.9.5 Erstellen von Ausgabedateien

Diese Anweisung dienen zum Erstellen von Ausgabedateien mit Hilfe der Codegenerierung.

Beachten Sie, dass all diese Anweisungen sich innerhalb eines Blocks befinden müssen, der innerhalb von eckigen Klammern stehen muss.

#### **create** *filename*

Erstellt eine neue Datei. Die Datei muss mit der **close**-Anweisung geschlossen werden. Die gesamte darauf folgende Ausgabe wird in die angegebene Datei geschrieben.

Beispiel:

```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name &
".java"]
package [= $JavaPackageName];

public class [= $application.Name]Application {
 ...
}
[close]
```

#### **close**

Schließt die aktuelle Ausgabedatei.

#### **=***\$variable*

Schreibt den Wert der angegebenen Variable in die aktuelle Ausgabedatei.

Beispiel:

```
[$x = 20+3]
The result of your calculation is [= $x] - so have a nice day!

- Die Dateiausgabe lautet:
The result of your calculation is 23 - so have a nice day!
```

#### **write** *string*

Schreibt den String in die aktuelle Ausgabedatei.

Beispiel:

```
[write "C" & $name]
```

Dies kann auch geschrieben werden als:

```
C[= $name]
```

#### **filecopy** *source to target*

Kopiert die Quelldatei ohne jede Interpretation in die Zieldatei.

Beispiel:

```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

### 31.9.6 Operatoren

Operatoren in SPL funktionieren wie in den meisten anderen Programmiersprachen.

Liste von SPL-Operatoren absteigend nach Vorrangigkeit gereiht:

|       |                                                     |
|-------|-----------------------------------------------------|
| .     | Objekteigenschaft aufrufen                          |
| ( )   | Gruppierung eines Ausdrucks                         |
| true  | Boolesche Konstante "true"                          |
| false | Boolesche Konstante "false"                         |
| &     | String-Verkettung                                   |
| -     | Zeichen für negative Zahl                           |
| not   | Logische Negation                                   |
| *     | Multiplizieren                                      |
| /     | Dividieren                                          |
| %     | Modulo                                              |
| +     | Addieren                                            |
| -     | Subtrahieren                                        |
| <=    | Kleiner oder gleich                                 |
| <     | Kleiner als                                         |
| >=    | Größer oder gleich                                  |
| >     | Größer als                                          |
| =     | Gleich                                              |
| <>    | Nicht gleich                                        |
| and   | Logische Konjunktion (mit short circuit-Auswertung) |
| or    | Logische Disjunktion (mit short circuit-Auswertung) |
| =     | Zuweisung                                           |

### 31.9.7 Bedingungen

SPL erlaubt die Verwendung von Standard-"if"-Anweisungen. Die Syntax lautet wie folgt:

```
if condition
 statements
else
 statements
endif
```

or, without else:

```
if condition
 statements
endif
```

Beachten Sie bitte, dass für die Bedingung keine runden Klammern verwendet werden! Wenn in allen anderen Programmiersprachen werden Bedingungen mit logischen und Vergleichsoperatoren konstruiert.

#### Bedingungen:

Wie bei jeder anderen Programmiersprache, werden Bedingungen mit logischen und Vergleichsoperatoren erstellt:

Beispiel:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
 whatever you want ['inserts whatever you want, in the resulting file]
[endif]
```

#### Switch

SPL enthält auch eine Multiple Choice-Anweisung.

Syntax:

```
switch $variable
 case X:
 statements
 case Y:
 case Z:
 statements
 default:
 statements
endswitch
```

Die Case-Bezeichnungen müssen Konstanten oder Variablen sein.

Die switch-Anweisung in SPL fällt nicht durch die Cases (wie in C), daher wird auch keine "break"-Anweisung benötigt.



### 31.9.8 Collections und foreach

#### Collections und Iteratoren

Eine Collection enthält mehrere Objekte - wie ein gewöhnliches Array. Iteratoren lösen das Problem des Speicherns und Inkrementierens von Array Indizes beim Aufrufen von Objekten.

Syntax:

```
foreach iterator in collection
 statements
next
```

Beispiel:

```
[foreach $class in $classes
 if not $class.IsInternal
] class [=$class.Name];
 [endif
next]
```

Beispiel 2:

```
[foreach $i in 1 To 3
 write "// Step " & $i & "\n"
 ' Do some work
next]
```

Die erste Zeile:

**\$classes** ist das [globale Objekt](#) von allen Typen. \$classes ist eine Collection (Sammlung) von einzelnen Klassenobjekten.

**Foreach** iteriert durch alle Datenelemente in \$classes und führt für die Anweisung den Code der auf die Anweisung folgt, bis zur **nächsten** Anweisung aus.

In jeder Iteration wird **\$class** dem nächsten Klassenobjekt zugewiesen. Sie arbeiten einfach mit dem Klassenobjekt, anstelle `classes[i]->class->Name()`, zu verwenden, wie das in C++ der Fall wäre.

Alle Collection-Iteratoren haben die folgenden zusätzlichen Eigenschaften:

|         |                                                                                                  |
|---------|--------------------------------------------------------------------------------------------------|
| Index   | der aktuelle Index beginnend mit 0                                                               |
| IsFirst | true, wenn das aktuelle Objekt das erste der Collection ist (Index ist 0)                        |
| IsLast  | true, wenn das aktuelle Objekt das letzte einer Collection ist                                   |
| Current | das aktuelle Objekt (dies ist implizit, wenn es nicht definiert ist und kann weggelassen werden) |

Beispiel:

```
[foreach $enum in $facet.Enumeration
 if not $enum.IsFirst
], [
 endif
]"[$enum.Value]"[
next]
```

#### Routinen zur Bearbeitung von Collections:

collection **SortByName**( bAscending )

gibt eine Collection zurück, deren Elemente in aufsteigender oder absteigender Reihenfolge (unter Berücksichtigung der Groß- und Kleinschreibung) nach Namen sortiert sind .

collection **SortByNameNoCase**( bAscending )  
gibt eine Collection zurück, deren Elemente in aufsteigender oder absteigender Reihenfolge (ohne Berücksichtigung der Groß- und Kleinschreibung) nach Namen sortiert sind .

Beispiel:

```
$SortedNestedClassifier = $Class.nestedClassifier.SortByNameNoCase(true)
```

collection **SortByKind**( bAscending )  
gibt eine Collection zurück, deren Elemente in aufsteigender oder absteigender Reihenfolge nach kind-Namen (z.B. "Class", "Interface",...) sortiert sind.

collection **SortByKindAndName**( bAscendingKind, bAscendingName )  
gibt eine Collection zurück, deren Elemente in aufsteigender oder absteigender Reihenfolge nach "kind" (z.B. "Class", "Interface",...) sortiert sind, wobei berücksichtigt wird, ob die "kinds" sortiert in aufsteigender oder absteigender Reihenfolge (unter Berücksichtigung der Groß- und Kleinschreibung) im Namen identisch sind.

collection **SortByKindAndNameNoCase**( bAscending )  
gibt eine Collection zurück, deren Elemente in aufsteigender oder absteigender Reihenfolge nach "kind" (z.B. "Class", "Interface",...) sortiert sind, wobei berücksichtigt wird, ob die "kinds" sortiert in aufsteigender oder absteigender Reihenfolge (ohne Berücksichtigung der Groß- und Kleinschreibung) im Namen identisch sind.

### 31.9.9 Subroutinen

Code Generator unterstützt nun Subroutinen in der Form von Prozeduren oder Funktionen.

Funktionen:

- Übergabe von Werten nach Wert und nach Referenz
- Lokale/globale Parameter (lokal innerhalb von Subroutinen)
- Lokale Variablen
- Rekursiver Aufruf (Subroutinen können sich selbst aufrufen)

#### Deklaration einer Subroutine

##### Subroutinen

Syntax-Beispiel:

```
Sub SimpleSub()
... lines of code
EndSub
```

- **Sub** ist das Schlüsselwort, das die Prozedur notiert.
- **SimpleSub** ist der Name, der der Subroutine zugewiesen wurde.
- Runde **Klammern** können eine Parameterliste enthalten.
- Der Code-Block einer Subroutine beginnt direkt nach der geschlossenen Klammer nach dem Parameter.
- **EndSub** notiert das Ende des Code-Blocks.

Bitte beachten Sie:

Eine rekursive oder kaskadierende Subroutine-**Deklaration** ist nicht zulässig, d.h. eine Subroutine darf keine weiteren Subroutinen enthalten.

##### Parameter

Parameter können auch durch Prozeduren unter Verwendung der folgenden Syntax übergeben werden:

- Alle Parameter müssen Variablen sein
- Variablen muss das **\$**-Zeichen vorangestellt werden
- Lokale Variablen werden in einer Subroutine definiert
- Globale Variablen werden explizit außerhalb von Subroutinen deklariert
- Mehrere Parameter werden innerhalb runder Klammern durch Kommas "," voneinander getrennt
- Parameter können Werte übergeben

#### Parameter - Übergabe von Werten

Parameter können auf zwei Arten übergeben werden: nach Wert oder nach Referenz, wobei die Schlüsselwörter **ByVal** bzw. **ByRef** verwendet werden.

Syntax:

```
' define sub CompleteSub()
[Sub CompleteSub($param, ByVal $paramByValue, ByRef $paramByRef)
] ...
```

- **ByVal** definiert, dass der Parameter nach Wert übergeben wird. Beachten Sie, dass die meisten Objekte nur nach Referenz übergeben werden können.
- **ByRef** definiert, dass der Parameter nach Referenz übergeben wird. Dies ist die Standardeinstellung, wenn weder ByVal noch ByRef definiert ist.

### Funktionsrückgabewerte

Für die Rückgabe eines Werts von einer Subroutine verwenden Sie ein **Return**-Statement. Eine solche Funktion kann von innerhalb eines Ausdrucks aufgerufen werden.

Beispiel:

```
' define a function
[Sub MakeQualifiedName(ByVal $namespacePrefix, ByVal $localName)
if $namespacePrefix = ""
 return $localName
else
 return $namespacePrefix & ":" & $localName
endif
EndSub
]
```

### Subroutinenaufruf

Verwenden Sie zum Aufrufen einer Subroutine **call**, gefolgt vom Namen der Prozedur und ggf. den Parametern.

```
Call SimpleSub()
```

oder,

```
Call CompleteSub("FirstParameter", $ParamByValue, $ParamByRef)
```

### Funktionsaufruf

Um eine Funktion (jede Subroutine, die ein **Return**-Statement enthält) aufzurufen, verwenden Sie einfach deren Namen in einem Ausdruck. Verwenden Sie zum Aufrufen von Funktionen nicht das **Call**-Statement.

Beispiel:

```
$QName = MakeQualifiedName($namespace, "entry")
```

### Beispiel für eine Subroutine

Das markierte Beispiel enthält die Deklaration und den Aufruf einer Subroutine.

```
A sample SPL file:
[create $outputpath & $module & "output.txt"

' define sub SimpleSub()
sub SimpleSub()
]SimpleSub() called
[endsub

' execute sub SimpleSub()
call SimpleSub()

$ParamByValue = "Original Value"
]ParamByValue = [= $ParamByValue]
[$ParamByRef = "Original Value"
]ParamByRef = [= $ParamByRef]

' define sub CompleteSub()
[sub CompleteSub($param, byval $paramByValue, byref $paramByRef)
]CompleteSub called.
 param = [= $param]
 paramByValue = [= $paramByValue]
 paramByRef = [= $paramByRef]
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
] Set values inside sub
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

' run sub CompleteSub()
call CompleteSub("FirstParameter", $ParamByValue, $ParamByRef)
]
ParamByValue=[= $ParamByValue]
ParamByRef=[= $ParamByRef]
[
close
]
```

### Derselbe Beispielcode:

```
[create $outputpath & $module & "output.txt"

' define sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

' execute sub SimpleSub()
Call SimpleSub()

$ParamByValue = "Original Value"
]ParamByValue = [= $ParamByValue]
[$ParamByRef = "Original Value"
]ParamByRef = [= $ParamByRef]

' define sub CompleteSub()
[Sub CompleteSub($param, ByVal $paramByValue, ByRef $paramByRef)
```

```
]CompleteSub called.
 param = [= $param]
 paramByValue = [= $paramByValue]
 paramByRef = [= $paramByRef]
 [$ParamByRef = "Local Variable"
 $paramByValue = "new value"
 $paramByRef = "new value"
] Set values inside Sub
 [$ParamByRef = "Local Variable"
 $paramByValue = "new value"
 $paramByRef = "new value"
]CompleteSub finished.
]endsub

' run sub CompleteSub()
Call CompleteSub("FirstParameter", $ParamByValue, $ParamByRef)
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
Close
]
```

### 31.9.10 Integrierte Typen

In diesem Abschnitt werden die in [vordefinierten Variablen](#) verwendeten Eigenschaften von integrierten Typen beschrieben, die das geparste Schema beschreiben.

#### Bibliothek

Dieses Objekt steht für die gesamte anhand des XML-Schemas oder der DTD generierte Bibliothek.

| Eigenschaft      | Typ                                  | Beschreibung                                                            |
|------------------|--------------------------------------|-------------------------------------------------------------------------|
| SchemaNamespaces | <a href="#">Namespace</a> collection | Namespaces in dieser Bibliothek                                         |
| SchemaFilename   | string                               | Name der XSD- oder DTD-Datei, von der diese Bibliothek abgeleitet wurde |
| SchemaType       | integer                              | 1 für DTD, 2 für XML-Schema                                             |
| Guid             | string                               | Eine globale eindeutige ID                                              |
| CodeName         | string                               | Generierter Bibliotheksname (vom Schemadateinamen abgeleitet)           |

#### Namespace

Pro XML-Schema Namespace wird ein Namespace-Objekt generiert. Schema-Komponenten, die sich in keinem Namespace befinden, werden in ein spezielles Namespace-Objekt mit einer leeren NamespaceURI gelegt.

Beachten Sie: Bei DTD werden Namespaces auch von Attributen abgeleitet, deren Namen mit "xmlns" beginnen.

| Eigenschaft  | Typ                             | Beschreibung                                          |
|--------------|---------------------------------|-------------------------------------------------------|
| CodeName     | String                          | Name für den generierten Code (abgeleitet vom Präfix) |
| LocalName    | String                          | Namespace-Präfix                                      |
| NamespaceURI | String                          | Namespace URI                                         |
| Types        | <a href="#">Type</a> Collection | Alle in diesem Namespace enthaltenen Typen            |
| Library      | <a href="#">Library</a>         | Bibliothek, die diesen Namespace enthält              |

#### Typ

Dieses Objekt stellt einen complex oder simpleType dar. Es dient zum Generieren einer Klasse in der Zielsprache.

Es gibt keinen zusätzlichen Typ pro Bibliothek, der für das Dokument steht, der alle möglichen Root-Elemente als Members hat.

Anonyme Typen haben einen leeren LocalName.

| Eigenschaft  | Typ                               | Beschreibung                                                                              |
|--------------|-----------------------------------|-------------------------------------------------------------------------------------------|
| CodeName     | string                            | Name für generierten Code (wird vom lokalen Namen oder der Parent-Deklaration abgeleitet) |
| LocalName    | string                            | Ursprünglicher Name im Schema                                                             |
| Namespace    | <a href="#">Namespace</a>         | Namespace, der diesen Typ enthält                                                         |
| Attributes   | <a href="#">Member</a> collection | Attribute, die in diesem Typ enthalten sind                                               |
| Elements     | <a href="#">Member</a> collection | Child-Elemente, die in diesem Typ enthalten sind                                          |
| IsSimpleType | boolean                           | True bei simpleTypes, False bei complexTypes                                              |

|                        |                         |                                                                                                                  |
|------------------------|-------------------------|------------------------------------------------------------------------------------------------------------------|
| IsDerived              | boolean                 | True, wenn dieser Typ von einem anderen Typ abgeleitet wurde, der ebenfalls durch ein Typobjekt dargestellt wird |
| IsDerivedByExtension   | boolean                 | "true", wenn dieser Typ mittels extension abgeleitet wurde                                                       |
| IsDerivedByRestriction | boolean                 | "true", wenn dieser Typ mittels restriction abgeleitet wurde                                                     |
| IsDerivedByUnion       | boolean                 | "true", wenn dieser Typ mittels union abgeleitet wurde                                                           |
| IsDerivedByList        | boolean                 | "true", wenn dieser Typ mittels list abgeleitet wurde                                                            |
| BaseType               | Type                    | Der Basistyp dieses Typs (wenn IsDerived "true" ist)                                                             |
| IsDocumentRootType     | boolean                 | True, wenn dieser Typ für das Dokument selbst steht                                                              |
| Library                | <a href="#">Library</a> | Bibliothek, die diesen Typ enthält                                                                               |
| IsFinal                | boolean                 | True, wenn als "final" im Schema deklariert                                                                      |
| IsMixed                | boolean                 | True, wenn dieser Typ Mixed Content haben kann                                                                   |
| IsAbstract             | boolean                 | True, wenn dieser Typ als abstrakt deklariert ist                                                                |
| IsGlobal               | boolean                 | True, wenn dieser Typ im Schema global deklariert ist                                                            |
| IsAnonymous            | boolean                 | True, wenn dieser Typ lokal in einem Element deklariert ist                                                      |

Nur bei simpleTypes:

| Eigenschaft   | Typ                           | Beschreibung                                 |
|---------------|-------------------------------|----------------------------------------------|
| IsNativeBound | boolean                       | True, wenn Native Type Binding vorhanden ist |
| NativeBinding | <a href="#">NativeBinding</a> | Native Binding für diesen Typ                |
| Facets        | <a href="#">Facets</a>        | Facets dieses Typs                           |
| Whitespace    | string                        | Shortcut zur Whitespace Facet                |

\* ComplexTypes mit Textinhalt (dabei handelt es sich um Typen mit Mixed Content und complexType mit simpleContent) haben ein zusätzliches nicht benanntes Attribut-Member, das den Textinhalt repräsentiert.

### Member

Dieses Objekt steht für ein Attribut oder Element im XML-Schema. Es dient zum Erstellen von Klassen-Members von Typen.

| Eigenschaft  | Typ    | Beschreibung                                                                                                  |
|--------------|--------|---------------------------------------------------------------------------------------------------------------|
| CodeName     | string | Name für generierten Code (vom lokalen Namen oder der Parent-Deklaration abgeleitet)                          |
| LocalName    | string | Ursprünglicher Name im Schema. Leer bei dem speziellen Member, das den Textinhalt von complexTypes darstellt. |
| NamespaceURI | string | Die Namespace URI dieses Elements/Attributs in XML-Instanzdokumenten / -streams.                              |



|                |                         |                                                                                                    |
|----------------|-------------------------|----------------------------------------------------------------------------------------------------|
| DeclaringType  | <a href="#">Type</a>    | Typ, der ursprünglich das Member deklariert (entspricht ContainingType für nicht vererbte Members) |
| ContainingType | <a href="#">Type</a>    | Typ, dessen Member dies ist                                                                        |
| DataType       | <a href="#">Type</a>    | Datentyp des Inhalts dieses Members                                                                |
| Library        | <a href="#">Library</a> | Bibliothek, die den Datentyp dieses Members enthält                                                |
| IsAttribute    | boolean                 | True bei Attributen, False bei Elementen                                                           |
| IsOptional     | boolean                 | True, wenn minOccurs = 0 oder optionales Attribut                                                  |
| IsRequired     | boolean                 | True, wenn minOccurs > 0 oder required Attribut                                                    |
| IsFixed        | boolean                 | True bei fixed Attributen, Wert ist in der Eigenschaft "Default"                                   |
| IsDefault      | boolean                 | True bei Attributen mit Standardwert, Wert ist in der Eigenschaft "Default"                        |
| IsNullible     | boolean                 | True bei nillable Elementen                                                                        |
| IsUseQualified | boolean                 | True, wenn NamespaceURI nicht leer ist                                                             |
| MinOccurs      | integer                 | minOccurs wie in Schema. 1 bei required Attributen                                                 |
| MaxOccurs      | integer                 | maxOccurs, wie in Schema. 0 bei prohibited Attributen, -1 bei unbounded                            |
| Default        | string                  | Standardwert                                                                                       |

### NativeBinding

Dieses Objekt steht für das Binding eines simpleType an einen native Type in der Zielprogrammiersprache, wie in der "schemanativetype" Zuordnung angegeben.

| Eigenschaft  | Typ    | Beschreibung                |
|--------------|--------|-----------------------------|
| ValueType    | string | Native Type                 |
| ValueHandler | string | Formatierungsklasseninstanz |

### Facets

Dieses Objekt steht für alle Facets eines simpleType.  
Vererbte Facets werden mit den explizit deklarierten Facets zusammengeführt. Wenn ein Length Facet gilt, werden MinLength und MaxLength auf denselben Wert gesetzt.

| Eigenschaft    | Typ     | Beschreibung                          |
|----------------|---------|---------------------------------------|
| DeclaringType  | Type    | Type Facets werden darauf deklariert  |
| Whitespace     | string  | "preserve", "collapse" oder "replace" |
| MinLength      | integer | Facet-Wert                            |
| MaxLength      | integer | Facet-Wert                            |
| MinInclusive   | integer | Facet-Wert                            |
| MinExclusive   | integer | Facet-Wert                            |
| MaxInclusive   | integer | Facet-Wert                            |
| MaxExclusive   | integer | Facet-Wert                            |
| TotalDigits    | integer | Facet-Wert                            |
| FractionDigits | integer | Facet-Wert                            |

|      |                  |                       |
|------|------------------|-----------------------|
| List | Facet collection | Alle Facets als Liste |
|------|------------------|-----------------------|

### Facet

Dieses Objekt steht für eine einzelne Facet, deren berechneter Wert für einen bestimmten Typ wirksam wird.

| Eigenschaft      | Typ                  | Beschreibung                                                                                                         |
|------------------|----------------------|----------------------------------------------------------------------------------------------------------------------|
| LocalName        | string               | Facet-Name                                                                                                           |
| NamespaceURI     | string               | Facet Namespace                                                                                                      |
| FacetType        | string               | Einer der folgenden: "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" oder "valuespace-range" |
| DeclaringType    | <a href="#">Type</a> | Typ, auf den dieses Facet deklariert ist                                                                             |
| FacetCheckerName | string               | Name des Facet Checker (aus der schemafacet-Zuordnung)                                                               |
| FacetValue       | string or integer    | Tatsächlicher Wert dieses Facet                                                                                      |

### Altes Objektmodell (bis v2007)

Die folgenden Objekte gehören dem alten Objektmodell für Codekompatibilität bis Version 2007 an. Sie werden in zukünftigen Versionen entfernt.

#### Namespace

Namespace-Abstraktion. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

| Eigenschaft           | Typ                              | Beschreibung                                                                            |
|-----------------------|----------------------------------|-----------------------------------------------------------------------------------------|
| URI                   | string                           | Die URI dieses Namespace                                                                |
| Prefix                | string                           | Das Präfix dieses Namespace                                                             |
| ContainsPublicClasses | boolean                          | True, wenn die Klassensammlung mindestens ein nicht-internes Klassenobjekt enthält.     |
| Classes               | <a href="#">Class</a> collection | Sammlung von Klassen in diesem Namespace - iterieren Sie mit <b>foreach</b> durch diese |

#### Klasse

Für jeden im Schema (xsd) deklarierten benutzerdefinierten Typ wird eine Klasse generiert. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

| Eigenschaft     | Typ                       | Beschreibung                                               |
|-----------------|---------------------------|------------------------------------------------------------|
| HasNamespace    | boolean                   | True, wenn es ein verknüpftes Namespace-Objekt gibt.       |
| Namespace       | <a href="#">Namespace</a> | Das Namespace-Objekt, dem dieses Klassenobjekt angehört.   |
| NamespacePrefix | string                    | Das Präfix des Namespace, in dem sich die Klasse befindet. |
| NamespaceURI    | string                    | URI des Namespace, in dem sich die Klasse befindet.        |

|                         |                                      |                                                                                                                              |
|-------------------------|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Name                    | string                               | Name des Typs der erzeugten Datei                                                                                            |
| SchemaName              | string                               | Name des Typs in der Original-Schema-Datei                                                                                   |
| HasBaseObject           | boolean                              | True, wenn dieser Typ von einem anderen Typ abgeleitet wurde, der auch durch ein Klassenobjekt dargestellt wird.             |
| BaseObject              | <a href="#">Class</a>                | Das Basisklassenobjekt, wenn HasBaseObject "true" ist.                                                                       |
| BaseNamespaceURI        | string                               | Namespace URI der Basisklasse                                                                                                |
| Base                    | string                               | Name der Basisklasse                                                                                                         |
| SchemaBase              | string                               | Name des Basistyps in der Original-Schema-Datei                                                                              |
| BuiltInBase             | string                               | Name des Root Simple Type                                                                                                    |
| BuiltInSchemaBase       | string                               | Name des Root Simple Type in der Original-Schema-Datei                                                                       |
| IsAbstract              | boolean                              | True, wenn ein abstract Type vorhanden ist.                                                                                  |
| IsSimpleType            | boolean                              | True, wenn dies ein simple Type ist.                                                                                         |
| IsComplexFromSimpleType | boolean                              | True, wenn dies ein complex Type ist, der von einem simple Type abgeleitet wurde.                                            |
| IsComplexType           | boolean                              | True, wenn ein complex Type vorhanden ist.                                                                                   |
| IsSequence              | boolean                              | True, wenn der Group-Type der obersten Ebene "sequence" ist.                                                                 |
| IsChoice                | boolean                              | True, wenn der Group-Type der obersten Ebene "choice" ist.                                                                   |
| IsAll                   | boolean                              | True, wenn der Group-Type der obersten Ebene "all" ist.                                                                      |
| Description             | string                               | Beschreibung dieses Typs. Kann Zeilenvorschubzeichen enthalten.                                                              |
| IsGlobal                | boolean                              | True, wenn dieser Typ im Schema global deklariert ist.                                                                       |
| IsAnonymous             | boolean                              | True, wenn dieser Typ im Schema lokal deklariert ist.                                                                        |
| IsInternal              | boolean                              | True, wenn es sich hierbei um die interne Root-Pseudoklasse handelt, die alle globalen Klassen als Members enthält.          |
| Members                 | <a href="#">Member</a><br>collection | Eine Klasse, die einen complex Type darstellt, enthält ein oder mehrere Members.                                             |
| Facets                  | <a href="#">Facet</a><br>collection  | Eine Klasse, die einen simpleType oder einen von einem simpleType abgeleiteten complexType darstellt, kann Facets enthalten. |

### Member

Abstraktion einer Member-Variable innerhalb einer Klasse. Members werden für alle Children eines complex Type erstellt. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

| Eigenschaft  | Typ    | Beschreibung                                                                             |
|--------------|--------|------------------------------------------------------------------------------------------|
| NamespaceURI | string | Die Namespace URI dieses Elements/Attributs innerhalb von XML-Instanzdokumenten/Streams. |
| Name         | string | Name in der erzeugten Datei                                                              |

|                         |                       |                                                                                                                                       |
|-------------------------|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| SchemaName              | string                | Name, wie er in der Original-Schema-Datei aufscheint.                                                                                 |
| XmlName                 | string                | Der Name, wie er in den XML-Instanzdokumenten/Streams aufscheinen soll.                                                               |
| Type                    | string                | Der Name der Klasse, die den Schematyp darstellt.                                                                                     |
| SchemaType              | string                | Der Schematyp.                                                                                                                        |
| TypeObject              | <a href="#">Class</a> | Die für diesen Typ generierte Klasse. Siehe Erklärung unten*                                                                          |
| HasTypeObject           | boolean               | True, wenn TypeObject einen gültigen Wert hat.                                                                                        |
| Description             | string                | Beschreibung des Elements/Attributs. Kann Zeilenvorschubzeichen enthalten.                                                            |
| IsBuiltInType           | boolean               | True, wenn der Typ ein integrierter Schematyp wie z.b. string, unsignedInt, dateTime... ist. Dies ist die Negation von HasTypeObject. |
| IsSimpleType            | boolean               | True, wenn der Typ dieses Member ein simpleType ist.                                                                                  |
| IsComplexFromSimpleType | boolean               | True, wenn der Typ dieses Member ein complex Type ist und von einem simple Type abgeleitet ist.                                       |
| IsElement               | boolean               | True, wenn dies ein Element ist.                                                                                                      |
| IsAttribute             | boolean               | True, wenn dies ein Attribut ist.                                                                                                     |
| NodeType                | string                | "Element" oder "Attribute"                                                                                                            |
| MinOcc                  | integer               | minOccurs, wie in Schema. 1 for obligatorische Attribute.                                                                             |
| MaxOcc                  | integer               | maxOccurs, wie in Schema. 0 für verbotene Attribute.                                                                                  |
| IsQualified             | boolean               | True, wenn die Form des Elements/Attributs auf "qualified" gesetzt ist.                                                               |
| IsMixed                 | boolean               | True, wenn dieses Element gemischten Inhalt haben kann.                                                                               |
| HasTextValue            | boolean               | Wenn dieses Member Text enthalten kann.                                                                                               |
| Default                 | string                | Der im Schema definierte Standardwert                                                                                                 |
| Fixed                   | string                | Der im Schema definierte "fixed" Wert.                                                                                                |

**\*TypeObject:**

Für jeden im Schema deklarierten Typ (xsd) wird eine Klasse generiert. Alle Elemente innerhalb eines complex Type werden als Member solcher Klassen generiert. Bei den Typen dieser Member kann es sich entweder um built-in Typen oder um benutzerdefinierte Typen handeln.

Built-in XML-Schematypen haben kein TypeObject, da sie mittels der [map type](#)-Anweisung auf vordefinierte Klassen gemappt werden. Für jedes Member mit einem benutzerdefinierten Typ stellt das TypeObject die für diesen Typ generierte Klasse dar.

Beispiel:

```
<xs:complexType name="TypeA">
 <xs:sequence>
```

```

 <xs:element name="B" type="TypeB"/>
 </xs:sequence>
</xs:complexType>
<xs:complexType name="TypeB"/>

```

CodeGen würde zwei Klassen erstellen:

```

class TypeA
{
 ...
 TypeB B;
}

class TypeB
{
 ...
}

```

Erklärung:

Wenn Klasse "TypeA" generiert wird, wird auch Member B generiert. Um den Typ B herauszufinden, verwenden Sie die "TypeObject" Methode des Elements, welches die Klasse "TypeB" zurückgibt. Auf diese Art können wir den Namen ermitteln und dort verwenden.

Facet

Diese Klasse besteht aus dem Constraint selbst und mehreren Booleschen Variablen. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

Die Booleschen Variablen geben an, um welche Art von Constraint es sich in dem Fall handelt. Die Namen der Booleschen Variablen sind mit den in der Schema-Spezifikation angegebenen identisch.

| Eigenschaft      | Typ                                    | Beschreibung                                                                                    |
|------------------|----------------------------------------|-------------------------------------------------------------------------------------------------|
| Constraint       | string                                 | Enthält den Wert des Facet.                                                                     |
| IsLength         | boolean                                |                                                                                                 |
| IsMinLength      | boolean                                |                                                                                                 |
| IsMaxLength      | boolean                                |                                                                                                 |
| IsMinInclusive   | boolean                                |                                                                                                 |
| IsMinExclusive   | boolean                                |                                                                                                 |
| IsMaxInclusive   | boolean                                |                                                                                                 |
| IsMaxExclusive   | boolean                                |                                                                                                 |
| IsTotalDigits    | boolean                                |                                                                                                 |
| IsFractionDigits | boolean                                |                                                                                                 |
| IsWhiteSpace     | boolean                                |                                                                                                 |
| IsPattern        | boolean                                |                                                                                                 |
| IsEnumeration    | boolean                                |                                                                                                 |
| Enumeration      | <a href="#">Enumeration</a> Collection | Enthält eine Sammlung von Enumerationsobjekten, wenn dieses Facet dem Typ Enumeration angehört. |
| Pattern          | <a href="#">Pattern</a> Collection     | Enthält eine Sammlung von Enumerationsobjekten, wenn dieses Facet dem Typ Pattern angehört.     |

### Enumeration

Abstraktion eines Enumerationseintrags innerhalb eines Facet. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

| Eigenschaft | Typ     | Beschreibung                                                 |
|-------------|---------|--------------------------------------------------------------|
| Index       | integer | Enthält den Index dieses Enumerationswerts, beginnend mit 1. |
| Value       | string  | Enthält einen Enumerationswert.                              |

### Pattern

Abstraktion eines Pattern-Eintrags innerhalb eines Facet. Dieses Objekt ist Teil des alten Objektmodells für die Codekompatibilität bis Version 2007.

| Eigenschaft | Typ     | Beschreibung                                           |
|-------------|---------|--------------------------------------------------------|
| Index       | Integer | Enthält den Index dieses Pattern-Werts beginnend mit 0 |
| Value       | String  | Enthält einen Pattern-Wert                             |

# Kapitel 32

---

## Die MapForce API

## 32 Die MapForce API

Über die COM-basierte API von MapForce können Clients die Funktionalitäten von MapForce ganz einfach aufrufen. Auf diese Art kann nun ein breites Spektrum an Aufgaben automatisiert werden.

In MapForce finden die allgemeinen von Microsoft vorgegebenen Spezifikationen für Automation Server Anwendung. Die Methoden und Eigenschaften der MapForce API können von allgemeinen Entwicklungsumgebungen wie solchen, in denen .NET, C++ und VisualBasic verwendet werden, und mit Scriptsprachen wie JavaScript und VBScript aufgerufen werden.

Es sollten in Ihrem Client Code die folgenden Richtlinien beachtet werden:

- Behalten Sie Referenzen auf Objekte nicht länger im Arbeitsspeicher, als notwendig. Wenn ein Benutzer zwischen zwei Calls Ihres Client eine Eingabe macht, besteht keine Garantie, dass diese Referenzen noch gültig sind.
- Denken Sie daran, dass bei einem Absturz Ihres Client Code Instanzen von MapForce möglicherweise noch im System verbleiben.
- Nähere Informationen, wie man lästige Fehlermeldungen vermeidet, finden Sie unter [Behandlung von Fehlern](#).
- Geben Sie Referenzen explizit frei, wenn Sie Sprachen wie C++ verwenden.

Um Ihre Version von MapForce2012 in die Microsoft Visual Studio IDE-Versionen 2005, 2008 und 2010 zu integrieren, gehen Sie vor wie folgt:

- Installieren Sie Microsoft Visual Studio
- Installieren Sie MapForce (Enterprise oder Professional Edition)
- Laden Sie das MapForce Visual Studio .NET Edition Integrationspaket für Microsoft Visual Studio .NET herunter und führen Sie es aus. Dieses Paket steht unter [www.altova.com](http://www.altova.com) auf der MapForce (Enterprise und Professional Edition) Download-Seite zur Verfügung.
- Die Beispieldateien für das Integrationspaket werden unterhalb des **c:\Program Files\Altova\MapForce2012\Examples\ActiveX\** Installationsordners für die entsprechenden Sprachen installiert.

### **Mapforce-Integration und Verwendung auf Client-Computern:**

Wenn Sie eine .NET Applikation erstellen und sie auf anderen Clients verwenden möchten, müssen Sie die folgenden Komponenten auf den Client Computern installieren:

- die MapForce Applikation
- das MapForce Integrationspaket
- den benutzerdefinierten Integrationscode.



## 32.1 Übersicht

In dieser Übersicht über die MapForce API finden Sie das Objektmodell für die API und eine Beschreibung der wichtigsten API-Konzepte. Die folgenden Themen werden behandelt:

- [Das Objektmodell](#)
- [Beispiel: Codegenerierung](#)
- [Beispiel: Mapping-Ausführung](#)
  - [Beispiel: Projektunterstützung](#)
  - [Behandlung von Fehlern](#)

### 32.1.1 Objektmodell

Der Ausgangspunkt für jede Applikation, die die MapForce API verwendet, ist das [Application](#) Objekt.

Um eine Instanz des Application Objekts zu erstellen, rufen Sie `CreateObject("MapForce.Application")` aus VisualBasic oder eine ähnliche Funktion aus der Entwicklungsumgebung Ihrer Wahl auf, um ein COM-Objekt zu erstellen. Um die vollständige MapForce API verwenden zu können, müssen keine anderen Objekte erstellt werden. Alle anderen Schnittstellen werden über andere Objekte aufgerufen, wobei das Application Objekt den Ausgangspunkt bildet.

Das Application-Objekt besteht aus den folgenden Teilen (jede Einrückung zeigt an, dass es sich dabei um eine Child-Parent-Beziehung zur Ebene unmittelbar oberhalb davon handelt):

- [Application](#)
  - [Options](#)
  - [Project](#)
    - [ProjectItem](#)
  - [Documents](#)
    - [Document](#)
      - [MapForceView](#)
      - [Mapping](#)
        - [Component](#)
          - [Datapoint](#)
        - [Components](#)
        - [Connection](#)
      - [Mappings](#)
    - [ErrorMarkers](#)
      - [ErrorMarker](#)
  - [AppOutputLines](#)
    - [AppOutputLine](#)
      - AppOutputLines
        - ...
        - [AppOutputLineSymbol](#)

Nachdem Sie ein Application Objekt erstellt haben, können Sie beginnen, die Funktionalitäten von MapForce zu verwenden. Im Allgemeinen öffnen Sie entweder ein bestehendes Document, erstellen ein neues oder generieren für oder anhand dieses Dokuments Code.

### 32.1.2 Beispiel: Codegenerierung

**Siehe auch**  
Codegenerierung

Im folgenden JScript-Beispiel wird gezeigt, wie man ein vorhandenes Dokument lädt und verschiedene Arten von Mapping-Code dafür generiert.

```
// ----- begin JScript example -----
// Generate Code for existing mapping.
// works with Windows scripting host.

// ----- helper function -----
function Exit(strErrorText)
{
 WScript.Echo(strErrorText);
 WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
 if (objErr != null)
 Exit ("ERROR: (" + (objErr.number & 0xffff) + ") " +
objErr.description + " - " + strText);
 else
 Exit ("ERROR: " + strText);
}
// -----

// ----- MAIN -----

// ----- create the Shell and FileSystemObject of the windows scripting
try
{
 objWshShell = WScript.CreateObject("WScript.Shell");
 objFSO = WScript.CreateObject("Scripting.FileSystemObject");
}
catch(err)
{ Exit("Can't create WScript.Shell object"); }

// ----- open MapForce or access running instance and make it visible
try
{
 objMapForce = WScript.GetObject ("", "MapForce.Application");
 objMapForce.Visible = true; // remove this line to perform
background processing
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }

// ----- open an existing mapping. adapt this to your needs!
objMapForce.OpenDocument(objFSO.GetAbsolutePathName ("Test.mfd"));

// ----- access the mapping to have access to the code generation methods
var objDoc = objMapForce.ActiveDocument;

// ----- set the code generation output properties and call the code
generation methods.
// ----- adapt the output directories to your needs
try
{
 // ----- code generation uses some of these options
 var objOptions = objMapForce.Options;

 // ----- generate XSLT -----
}
```

```
 objOptions.XSLTDefaultOutputDirectory = "C:\\test\\TestCOMServer\\XSLT"
 ;
 objDoc.GenerateXSLT();

 // ----- generate Java Code -----
 objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\Java"
 ;
 objDoc.GenerateJavaCode();

 // ----- generate CPP Code, use same cpp code options as the last time

 objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CPP";
 objDoc.GenerateCppCode();

 // ----- generate C# Code, use options C# code options as the last time

 objOptions.CodeDefaultOutputDirectory =
"C:\\test\\TestCOMServer\\CHash";
 objDoc.GenerateCHashCode();
}
catch (err)
{ ERROR ("while generating XSL or program code", err); }

// hide MapForce to allow it to shut down
objMapForce.Visible = false;

// ----- end example -----
```

### 32.1.3 Beispiel: Mapping-Ausführung

Im folgenden JScript-Beispiel wird gezeigt, wie Sie ein bestehendes Dokument mit einem einfachen Mapping laden, seine Komponenten aufrufen, Input- und Output-Instanzdateinamen definieren und das Mapping ausführen.

```

/*
 This sample file performs the following operations:

 Load existing MapForce mapping document.
 Find source and target component.
 Set input and output instance filenames.
 Execute the transformation.

 Works with Windows scripting host.
*/

// ---- general helpers -----

function Exit(message)
{
 WScript.Echo(message);
 WScript.Quit(-1);
}

function ERROR(message, err)
{
 if(err != null)
 Exit("ERROR: (" + (err.number & 0xffff) + ") " + err.description + " - " + message);
 else
 Exit("ERROR: " + message);
}

// ---- MapForce constants -----

var eComponentUsageKind_Unknown = 0;
var eComponentUsageKind_Instance = 1;
var eComponentUsageKind_Input = 2;
var eComponentUsageKind_Output = 3;

// ---- MapForce helpers -----

// Searches in the specified mapping for a component by name and returns it.
// If not found, throws an error.
function FindComponent(mapping, component_name)
{
 var components = mapping.Components;
 for(var i = 0 ; i < components.Count ; ++i)
 {
 var component = components.Item(i + 1);
 if(component.Name == component_name)
 return component;
 }
 throw new Error("Cannot find component with name " + component_name);
}

// Browses components in a mapping and returns the first one found acting as

```

```

// source component (i.e. having connections on its right side).
function GetFirstSourceComponent(mapping)
{
 var components = mapping.Components;
 for(var i = 0 ; i < components.Count ; ++i)
 {
 var component = components.Item(i + 1);
 if(component.UsageKind == eComponentUsageKind_Instance &&
 component.HasOutgoingConnections)
 {
 return component;
 }
 }
 throw new Error("Cannot find a source component");
}

// Browses components in a mapping and returns the first one found acting as
// target component (i.e. having connections on its left side).
function GetFirstTargetComponent(mapping)
{
 var components = mapping.Components;
 for(var i = 0 ; i < components.Count ; ++i)
 {
 var component = components.Item(i + 1);
 if(component.UsageKind == eComponentUsageKind_Instance &&
 component.HasIncomingConnections)
 {
 return component;
 }
 }
 throw new Error("Cannot find a target component");
}

function IndentTextLines(s)
{
 return "\t" + s.replace(/\n/g, "\n\t");
}

function GetAppoutputLineFullText(oAppoutputLine)
{
 var s = oAppoutputLine.GetLineText();
 var oAppoutputChildLines = oAppoutputLine.ChildLines;
 var i;

 for(i = 0 ; i < oAppoutputChildLines.Count ; ++i)
 {
 oAppoutputChildLine = oAppoutputChildLines.Item(i + 1);
 sChilds = GetAppoutputLineFullText(oAppoutputChildLine);
 s += "\n" + IndentTextLines(sChilds);
 }

 return s;
}

// Create a nicely formatted string from AppOutputLines
function GetResultMessagesString(oAppoutputLines)
{
 var s1 = "Transformation result messages:\n";
 var oAppoutputLine;
 var i;

 for(i = 0 ; i < oAppoutputLines.Count ; ++i)
 {

```

```

 oAppoutputLine = oAppoutputLines.Item(i + 1);
 s1 += GetAppoutputLineFullText(oAppoutputLine);
 s1 += "\n";
 }

 return s1;
}

// ---- MAIN -----

var wshShell;
var fso;
var mapforce;

// create the Shell and FileSystemObject of the windows scripting system
try
{
 wshShell = WScript.CreateObject("WScript.Shell");
 fso = WScript.CreateObject("Scripting.FileSystemObject");
}
catch(err)
{ ERROR("Can't create windows scripting objects", err); }

// open MapForce or access currently running instance
try
{
 mapforce = WScript.GetObject("", "MapForce.Application");
}
catch(err)
{ ERROR("Can't access or create MapForce.Application", err); }

try
{
 // open an existing mapping.
 // **** adjust the examples path to your needs ! *****
 var sMapForceExamplesPath = fso.BuildPath(
 wshShell.SpecialFolders("MyDocuments"),
 "Altova\MapForce2012\MapForceExamples");
 var sDocFilename = fso.BuildPath(sMapForceExamplesPath, "PersonList.mfd");
 var doc = mapforce.OpenDocument(sDocFilename);

 // Find existing components by name in the main mapping.
 // Note, the names of components may not be unique as a schema component's
 // name
 // is derived from its schema file name.
 var source_component = FindComponent(doc.MainMapping, "Employees");
 var target_component = FindComponent(doc.MainMapping, "PersonList");
 // If you do not know the names of the components for some reason, you
 // could
 // use the following functions instead of FindComponent.
 //var source_component = GetFirstSourceComponent(doc.MainMapping);
 //var target_component = GetFirstTargetComponent(doc.MainMapping);

 // specify the desired input and output files.
 source_component.InputInstanceFile = fso.BuildPath(sMapForceExamplesPath,
 "Employees.xml");
 target_component.OutputInstanceFile = fso.BuildPath(
 sMapForceExamplesPath, "test_transformation_results.xml");

 // Perform the transformation.
 // You can use doc.GenerateOutput() if you do not need result messages.
 // If you have a mapping with more than one target component and you want

```

```
// to execute the transformation only for one specific target component,
// call target_component.GenerateOutput() instead.
var result_messages = doc.GenerateOutputEx();

var summary_info =
 "Transformation performed from " + source_component.InputInstanceFile
+ "\n" +
 "to " + target_component.OutputInstanceFile + "\n\n" +
 GetResultMessagesString(result_messages);
 WScript.Echo(summary_info);
}
catch(err)
{
 ERROR("Failure", err);
}
```



### 32.1.4 Beispiel: Projektunterstützung

#### Siehe auch

Codegenerierung

Im folgenden JScript-Beispiel wird gezeigt, wie Sie das MapForce Projekt und Projektelement-Objekte der MapForce API zum Automatisieren komplexer Aufgaben verwenden können. Je nach Installation müssen Sie eventuell den Wert der Variable `strSamplePath` in den Beispielordner Ihrer MapForce Installation ändern.

Um alle Operationen im nachfolgenden Beispiel erfolgreich ausführen zu können, benötigen Sie die Enterprise Version von MapForce. Wenn Sie mit der Professional Version arbeiten, sollten Sie die Zeilen, in denen das Webservice-Projekt eingefügt wird, auskommentieren. Benutzer der Basic Edition haben überhaupt keinen Zugriff auf projektbezogene Funktionen.

```
// //////////// global variables ////////////
var objMapForce = null;
var objWshShell = null;
var objFSO = null;

// !!! adapt the following path to your needs. !!!
var strSamplePath = "C:\\Documents and Settings\\<username>\\My
Documents\\Altova\\MapForce2012\\MapForceExamples\\Tutorial\\";

// //////////// Helpers ////////////

function Exit(strErrorText)
{
 WScript.Echo(strErrorText);
 WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
 if (objErr != null)
 Exit ("ERROR: (" + (objErr.number & 0xffff) + ") " +
objErr.description + " - " + strText);
 else
 Exit ("ERROR: " + strText);
}

function CreateGlobalObjects ()
{
 // the Shell and FileSystemObject of the windows scripting host often
 always useful
 try
 {
 objWshShell = WScript.CreateObject("WScript.Shell");
 objFSO = WScript.CreateObject("Scripting.FileSystemObject");
 }
 catch(err)
 { Exit("Can't create WScript.Shell object"); }

 // create the MapForce connection
 // if there is a running instance of MapForce (that never had a
 connection) - use it
 // otherwise, we automatically create a new instance
 try
 {
 objMapForce = WScript.GetObject("", "MapForce.Application");
 }
 catch(err)
 {

```

```

 { Exit("Can't access or create MapForce.Application"); }
 }
}

// -----
// print project tree items and their properties recursively.
// -----
function PrintProjectTree(objProjectItemIter, strTab)
{
 while (! objProjectItemIter.atEnd())
 {
 // get current project item
 objItem = objProjectItemIter.item();

 try
 {
 // ----- print common properties
 strGlobalText += strTab + "[" + objItem.Kind + "]" +
objItem.Name + "\n";

 // ----- print code generation properties, if available
 try
 {
 if (objItem.CodeGenSettings_UseDefault)
 strGlobalText += strTab + " Use default
code generation settings\n";
 else
 strGlobalText += strTab + " code generation
language is " +
objItem.CodeGenSettings_Language +
objItem.CodeGenSettings_OutputFolder + "\n";
 }
 catch(err) {}

 // ----- print WSDL settings, if available
 try
 {
 strGlobalText += strTab + " WSDL File is " +
objItem.WSDLFile +
objItem.QualifiedName + "\n";
 }
 catch(err) {}
 }
 catch(ex)
 { strGlobalText += strTab + "[" + objItem.Kind + "]\n" }

 // ---- recurse
 PrintProjectTree(new Enumerator(objItem), strTab + ' ');
 objProjectItemIter.moveToNext();
 }
}

// -----
// Load example project installed with MapForce.
// -----
function LoadSampleProject()
{
 // close open project
 objProject = objMapForce.ActiveProject;
 if (objProject != null)
 objProject.Close();
}

```

```

 // open sample project and iterate through it.
 // sump properties of all project items

 objProject = objMapForce.OpenProject(strSamplePath +
"MapForceExamples.mfp");
 strGlobalText = '';
 PrintProjectTree(new Enumerator (objProject), ' ')
 WScript.Echo(strGlobalText);

 objProject.Close();
}

// -----
// Create a new project with some folders, mappings and a
// web service project.
// -----
function CreateNewProject()
{
 try
 {
 // create new project and specify file to store it.
 objProject = objMapForce.NewProject(strSamplePath + "Sample.mfp"
);

 // create a simple folder structure
 objProject.CreateFolder("New Folder 1");
 objFolder1 = objProject.Item(0);
 objFolder1.CreateFolder("New Folder 2");
 objFolder2 = (new Enumerator(objFolder1)).item(); // an
alternative to Item(0)

 // add two different mappings to folder structure
 objFolder1.AddFile(strSamplePath + "DB_Altova_SQLXML.mfd");
 objMapForce.Documents.OpenDocument(strSamplePath +
"InspectionReport.mfd");
 objFolder2.AddActiveFile();

 // override code generation settings for this folder
 objFolder2.CodeGenSettings_UseDefault = false;
 objFolder2.CodeGenSettings_OutputFolder = strSamplePath +
"SampleOutput"
 objFolder2.CodeGenSettings_Language = 1; //C++

 // insert web service project based on a wsdl file from the
installed examples
 objProject.InsertWebService(strSamplePath +
"TimeService/TimeService.wsdl",
"http://www.Nanonull.com/TimeService/}TimeService",
"TimeServiceSoap"
,
true);

 objProject.Save();
 if (! objProject.Saved)
 WScript.Echo("problem occurred when saving project");

 // dump project tree
 strGlobalText = '';
 PrintProjectTree(new Enumerator (objProject), ' ')
 WScript.Echo(strGlobalText);
 }
 catch (err)
 { ERROR("while creating new project", err); }
}

```

```
// -----
// Generate code for a project's sub-tree. Mix default code
// generation parameters and overloaded parameters.
// -----
function GenerateCodeForNewProject()
{
 // since the Web service project contains only initial mappings,
 // we generate code only for our custom folder.
 // code generation parameters from project are used for Folder1,
 // whereas Folder2 provides overwritten values.
 objFolder = objProject.Item(0);
 objFolder1.GenerateCode();
}

// ////////////////////////////////// MAIN //////////////////////////////////

CreateGlobalObjects();
objMapForce.Visible = true;

LoadSampleProject();
CreateNewProject();
GenerateCodeForNewProject();

// uncomment to shut down application when script ends
// objMapForce.Visible = false;
```

### 32.1.5 Behandlung von Fehlern

Die MapForce API gibt Fehler auf zwei verschiedene Arten zurück. Jede API-Methode gibt ein HRESULT zurück. Dieser Rückgabewert informiert den Caller über etwaige Fehlfunktionen während der Ausführung dieser Methode. Wenn der Call erfolgreich ausgeführt wurde, ist der Rückgabewert gleich S\_OK. C/C++-Programmierer verwenden im Allgemeinen HRESULT zum Ausfindigmachen von Fehlern.

VisualBasic, Script-Sprachen und andere komplexe Entwicklungsumgebungen geben dem Programmierer keinen Zugriff auf das zurückgegebene HRESULT eines COM Call. Sie verwenden den zweiten Mechanismus zur Auslösung von Fehlern, der von der MapForce API unterstützt wird, die IErrorInfo Schnittstelle. Wenn ein Fehler auftritt, erstellt die API ein neues Objekt, das die IErrorInfo Schnittstelle implementiert. Die Entwicklungsumgebung nimmt diese Schnittstelle und befüllt ihren Fehlerbehandlungsmechanismus mit den erhaltenen Informationen.

Im folgenden Text wird beschrieben, wie man von der MapForce API ausgelöste Fehler in verschiedenen Entwicklungsumgebungen behandelt.

#### VisualBasic

Eine häufige Methode zur Fehlerbehandlung in VisualBasic ist, einen Fehler-Handler zu definieren. Dieser Fehler-Handler kann mit der On Error Anweisung definiert werden. Normalerweise zeigt der Fehler-Handler eine Fehlermeldung an und bereinigt den Code um überflüssige Referenzen und alle Arten von Resouce Leaks zu vermeiden. VisualBasic befüllt sein eigenes Err Objekt mit den Informationen aus der IErrorInfo Schnittstelle.

Beispiel:

```
Sub Validate()
 'place variable declarations here

 'set error handler
 On Error GoTo ErrorHandler

 'if generation fails, program execution continues at ErrorHandler:
 objMapForce.ActiveDocument.GenerateXSLT()

 'additional code comes here

 'exit
Exit Sub

ErrorHandler:
 MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
 "Description: " & Err.Description)
End Sub
```

#### JavaScript

Die Microsoft Implementierung von JavaScript (JScript) bietet einen try-catch Mechanismus zur Behandlung von Fehlern, die von COM Calls ausgegeben werden. Er ähnelt der Methode von VisualBasic insofern, als ein Fehlerobjekt deklariert wird, das die nötigen Informationen enthält.

Beispiel:

```
function Generate()
{
 // please insert variable declarations here

 try
```

```
{
 objMapForce.ActiveDocument.GenerateXSLT();
}
catch(Error)
{
 sError = Error.description;
 nErrorCode = Error.number & 0xffff;
 return false;
}

return true;
}
```

### C/C++

C/C++ gibt Ihnen einfachen Zugriff auf das HRESULT des COM Call und das IErrorInterface.

```
HRESULT hr;

// Call GenerateXSLT() from the MapForce API
if(FAILED(hr = ipDocument->GenerateXSLT()))
{
 IErrorInfo *ipErrorInfo = Null;

 if(SUCCEEDED(::GetErrorInfo(0, &ipErrorInfo)))
 {
 BSTRbstrDescr;
 ipErrorInfo->GetDescription(&bstrDescr);

 // handle Error information
 wprintf(L"Error message:\t%s\n", bstrDescr);
 ::SysFreeString(bstrDescr);

 // release Error info
 ipErrorInfo->Release();
 }
}
```

### 32.1.6 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*). Die in diesem Abschnitt verwendeten Codefragmente 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/MapForce2012/MapForceExamples/API/ |
| Windows Vista, Windows 7 | C:/Users/<Benutzername>/Documents/Altova/MapForce2012/MapForceExamples/API/                            |

#### C#

Sie können mit Hilfe von C# auf die Funktionalitäten der Application API zugreifen. Anhand des aufgelisteten Codes wird gezeigt, wie Sie die API für bestimmte grundlegende Funktionalitäten aufrufen können.

#### Java

Sie können mit Hilfe von Java auf die Funktionalitäten der MapForce API zugreifen. In diesem Abschnitt wird erklärt, wie einige grundlegende MapForce-Funktionen über Java-Code aufgerufen werden können. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert.

#### JScrip

Im JScrip-Code werden die folgenden grundlegenden Funktionen verwendet.

#### **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 MapForce-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/MapForce2012/MapForceExamples/API/ |
|------------|--------------------------------------------------------------------------------------------------------|

|                          |                                                                             |
|--------------------------|-----------------------------------------------------------------------------|
| Windows Vista, Windows 7 | C:/Users/<Benutzername>/Documents/Altova/MapForce2012/MapForceExamples/API/ |
|--------------------------|-----------------------------------------------------------------------------|

Sie können das Projekt innerhalb von Visual Studio 2008 oder Visual Studio 2010 kompilieren und ausführen.

**Dateiliste:**

|                             |                                           |
|-----------------------------|-------------------------------------------|
| Readme.txt                  | Diese Datei                               |
| AutomateMapForce.csproj     | Visual Studio 2008- und 2010-Projektdatei |
| AutomateMapForce_VS2008.sln | Visual Studio 2008 Solution-Datei         |
| AutomateMapForce_VS2010.sln | Visual Studio 2010 Solution-Datei         |
| Program.cs                  | C# Beispiel Quellcode                     |

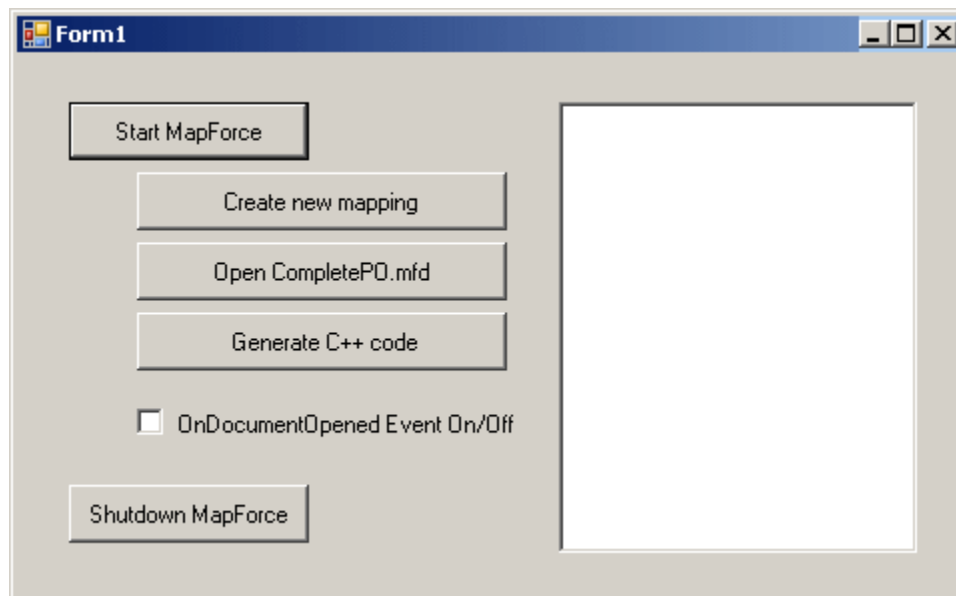
**Eigenschaften:**

Form1.cs  
Form1.Designer.cs  
Form1.resx

**Funktionen in diesem Beispiel**

In diesem Beispiel sehen Sie eine einfache Benutzeroberfläche (*Abbildung unten*) mit Schaltflächen, über die grundlegende MapForce-Operationen aufgerufen werden:

- Startet MapForce
- Erstellt ein neues Mapping-Design
- Öffnet die Datei CompletePO.mfd aus dem Ordner ...MapForceExamples
- Generiert C++ Code in einem temp-Verzeichnis
- Beendet MapForce

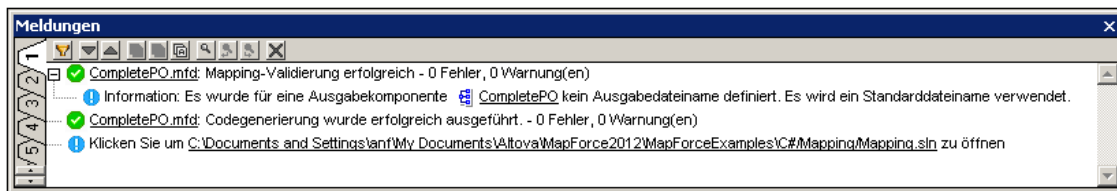




### Kompilieren und Ausführen des Beispiels

Doppelklicken Sie 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**.

Wenn Sie auf die Schaltfläche "C++ generieren" klicken, wird Code für das zuvor geöffnete Mapping CompletePO.mfd generiert.



### Code von Form1.cs

Zur besseren Erklärung enthält der Code Kommentare. 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.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication2
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();

 // An instance of MapForce accessed via its automation interface.
 MapForceLib.Application MapForce;

 // Location of examples installed with MapForce
 String strExamplesFolder;

 private void Form1_Load(object sender, EventArgs e)
 {
 // locate examples installed with MapForce.
 // REMARK: You might need to adapt this if you have a different
 // major version of the product.
 strExamplesFolder = Environment.GetEnvironmentVariable(
```

```

"USERPROFILE") + "\\My Documents\\Altova\\MapForce2012\\MapForceExamples\\";
 }

 // handler for the "Start MapForce" button
 private void StartMapForce_Click(object sender, EventArgs e)
 {
 if (MapForce == null)
 {
 Cursor.Current = Cursors.WaitCursor;

 // if we have no MapForce instance, we create one and make it
visible.
 MapForce = new MapForceLib.Application();
 MapForce.Visible = true;

 Cursor.Current = Cursors.Default;
 }
 else
 {
 // if we have already an MapForce instance running we toggle
its visibility flag.
 MapForce.Visible = !MapForce.Visible;
 }
 }

 // handler for the "Open CompletePO.mfd" button
 private void openCompletePO_Click(object sender, EventArgs e)
 {
 if (MapForce == null)
 StartMapForce_Click(null, null);

 // Open one of the sample files installed with the product.
 MapForce.OpenDocument(strExamplesFolder + "CompletePO.mfd");
 }

 // handler for the "Create new mapping" button
 private void newMapping_Click(object sender, EventArgs e)
 {
 if (MapForce == null)
 StartMapForce_Click(null, null);

 // Create a new mapping
 MapForce.NewMapping();
 }

 // handler for the "Shutdown MapForce" button
 // shut-down application instance by explicitly releasing the COM
object.
 private void shutdownMapForce_Click(object sender, EventArgs e)
 {
 if (MapForce != null)
 {
 // allow shut-down of MapForce by releasing UI
 MapForce.Visible = false;

 // explicitly release COM object
 try
 {
 while (System.Runtime.InteropServices.Marshal
.ReleaseComObject(MapForce) > 0) ;
 }
 finally
 {
 // avoid later access to this object.
 MapForce = null;
 }
 }
 }

```

```

 }
}

// handler for button "Generate C++ Code"
private void generateCppCode_Click(object sender, EventArgs e)
{
 if (MapForce == null)
 listBoxMessages.Items.Add("start MapForce first.");
 // COM errors get returned to C# as exceptions. We use a try/catch
 block to handle them.
 try
 {
 String strError = "";
 MapForceLib.Document doc = MapForce.ActiveDocument;

 listBoxMessages.Items.Add("Active document " + doc.Name);
 MapForceLib.ErrorMarkers errMarkers =
doc.GenerateCodeEx(MapForceLib.ENUMProgrammingLanguage.eCpp);

 System.Collections.IEnumerator errMarkersCollection =
errMarkers.GetEnumerator();

 bool bEmpty = true;
 while (errMarkersCollection.MoveNext())
 {
 bEmpty = false;
 Object obj = errMarkersCollection.Current;

 if (obj is MapForceLib.ErrorMarker)
 strError = ((MapForceLib.ErrorMarker)obj).Text;
 listBoxMessages.Items.Add("Error text: " + strError);
 }

 if (bEmpty)
 listBoxMessages.Items.Add("Code generation completed
successfully.");
 }
 catch (Exception ex)
 {
 // The COM call was not successful.
 // Probably no application instance has been started or no
document is open.
 MessageBox.Show("COM error: " + ex.Message);
 }
}

delegate void addListBoxItem_delegate(string sText);
// called from the UI thread
private void addListBoxItem(string sText)
{
 listBoxMessages.Items.Add(sText);
}
// wrapper method to allow to call UI controls methods from a worker
thread
void syncWithUiThread(Control ctrl, addListBoxItem_delegate
methodToInvoke, String sText)
{
 // Control.Invoke: Executes on the UI thread, but calling thread
waits for completion before continuing.
 // Control.BeginInvoke: Executes on the UI thread, and calling
thread doesn't wait for completion.
 if (ctrl.InvokeRequired)
 ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
}

// event handler for OnDocumentOpened event

```

```

private void handleOnDocumentOpened(MapForceLib.Document i_ipDocument)
{
 String sText = "";

 if (i_ipDocument.Name.Length > 0)
 sText = "Document " + i_ipDocument.Name + " was opened!";
 else
 sText = "A new mapping was created.";

 // we need to synchronize the calling thread with the UI thread
 // the COM events are triggered from a working thread
 addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
 // call syncWithUIThread with the following arguments:
 // 1 - listBoxMessages - list box control to display messages from
COM events
 // 2 - methodToInvoke - a C# delegate which points to the method
which will be called from the UI thread
 // 3 - sText - the text to be displayed in the list box
 syncWithUIThread(listBoxMessages, methodToInvoke, sText);
}

private void checkBoxEventOnOff_CheckedChanged(object sender,
EventArgs e)
{
 if (MapForce != null)
 {
 if (checkBoxEventOnOff.Checked)
 MapForce.OnDocumentOpened += new MapForceLib.
_IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
 else
 MapForce.OnDocumentOpened -= new MapForceLib.
_IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
 }
}
}
}

```

## Java

Sie können mit Hilfe von Java-Code auf die Applikations-API zugreifen. Damit Sie direkt vom Java-Code aus Zugriff auf den MapForce Automation Server haben, müssen sich die unten aufgelisteten Bibliotheken im classpath befinden. Sie sind im MapForce Applikationsordner im Ordner JavaAPI installiert.

- AltovaAutomation.dll: ein JNI Wrapper für Altova Automation Server
- AltovaAutomation.jar: Java-Klassen zum Aufrufen von Altova Automation Servern
- MapForceAPI.jar: Java-Klassen, die als Wrapper für die MapForce Automation-Schnittstelle dienen
- MapForceAPI\_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 modifizieren 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 MapForce 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

Im folgenden Abschnitt wird erklärt, wie einige grundlegende MapForce-Funktionen über Java-Code aufgerufen werden können.

- [Java-Beispielprojekt](#)

#### Java-Beispielprojekt

Das MapForce-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/MapForce2012/MapForceExamples/API/ |
| Windows Vista, Windows 7 | C:/Users/<Benutzername>/Documents/Altova/MapForce2012/MapForceExamples/API/                            |

Dieser Ordner enthält Java-Beispiele für die MapForce 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:

|                         |                                                                                                                                                                              |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AltovaAutomation.dll    | Java-COM Bridge: DLL-Teil                                                                                                                                                    |
| AltovaAutomation.jar    | Java-COM Bridge: Java-Bibliotheksteil                                                                                                                                        |
| XMLSpyAPI.jar           | Java-Klassen der MapForce API                                                                                                                                                |
| RunXMLSpy.java          | Java-Beispielquellcode                                                                                                                                                       |
| BuildAndRun.bat         | Batch-Datei zum Kompilieren und Ausführen des Beispielcodes über die Befehlszeile. Es wird ein Ordner benötigt, in dem sich die Java Virtual Machine als Parameter befindet. |
| .classpath              | Hilfdatei Eclipse-Projekt                                                                                                                                                    |
| .project                | Eclipse-Projektdatei                                                                                                                                                         |
| MapForceAPI_JavaDoc.zip | Javadoc Datei, die die Hilfedokumentation für die Java API enthält                                                                                                           |

### Funktionen in diesem Beispiel

In diesem Beispielprojekt wird MapForce gestartet und einige Operationen wie das Öffnen und Schließen von Dokumenten werden ausgeführt. MapForce bleibt danach geöffnet. Sie müssen die Applikation manuell schließen.

### 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 RunMapForce.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 RunMapForce 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 RunMapForce.java.

```
// access general JAVA-COM bridge classes
import java.util.Iterator;
```

```

import com.altova.automation.libs.*;

// access MapForce Java-COM bridge
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts MapForce COM server and performs a few
 * operations on it.
 * Feel free to extend.
 */
public class RunMapForce
{
 public static void main(String[] args)
 {
 // an instance of the application.
 Application mapforce = null;

 // instead of COM error handling use Java exception mechanism.
 try
 {
 // Start MapForce as COM server.
 mapforce = new Application();
 // COM servers start up invisible so we make it visible
 mapforce.setVisible(true);

 // The following lines attach to the application events using a
 default implementation
 // for the events and override one of its methods.
 // If you want to override all document events it is better to derive
 your listener class
 // from DocumentEvents and implement all methods of this interface.
 mapforce.addListener(new ApplicationEventsDefaultHandler()
 {
 @Override
 public void onDocumentOpened(Document i_ipDoc) throws
AutomationException
 {
 String name = i_ipDoc.getName();

 if (name.length() > 0)
 System.out.println("Document " + name + " was opened.");
 else
 System.out.println("A new mapping was created.");
 }
 });

 // Locate samples installed with the product.
 String strExamplesFolder = System.getenv("USERPROFILE") + "\\My
Documents\\Altova\\MapForce2012\\MapForceExamples\\";
 // create a new MapForce mapping and generate c++ code
 Document newDoc = mapforce.newMapping();
 ErrorMarkers err1 =
newDoc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
 display(err1);
 // open CompletePO.mfd and generate c++ code
 Document doc = mapforce.openDocument(strExamplesFolder +
"CompletePO.mfd");
 ErrorMarkers err2 = doc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
 display(err2);

 doc.close();
 doc = null;

 System.out.println("Watch MapForce!");
 }
 }
}

```

```

 catch (AutomationException e)
 {
 // e.printStackTrace();
 }
 finally
 {
 // Make sure that MapForce can shut down properly.
 if (mapforce != null)
 mapforce.dispose();

 // Since the COM server was made visible and still is visible, it
 // will keep running
 // and needs to be closed manually.
 System.out.println("Now close MapForce!");
 }
 }

 public static void display(ErrorMarkers err) throws AutomationException
 {
 Iterator<ErrorMarker> itr = err.iterator();

 if (err.getCount() == 0)
 System.out.print("Code generation completed successfully.\n");

 while (itr.hasNext())
 {
 String sError = "";
 Object element = itr.next();
 if (element instanceof ErrorMarker)
 sError = ((ErrorMarker)element).getText();
 System.out.print("Error text: " + sError + "\n");
 }
 }
}

```

### JScript

Dieser Abschnitt enthält eine Liste von JScript-Code, anhand dessen die folgenden grundlegenden Funktionalitäten erläutert werden:

### Beispieldateien

Sie finden den Code 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<br>Einstellungen/<Benutzername>/Eigene Dateien/<br>Altova/MapForce2012/MapForceExamples/API/ |
| Windows Vista, Windows<br>7 | C:/Users/<Benutzername>/Documents/<br>Altova/MapForce2012/MapForceExamples/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.

|                   |                                                                                                                                     |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| DocumentAccess.js | Hier wird gezeigt, wie man Dokumente <a href="#">öffnet</a> , durch diese iteriert und sie schließt.                                |
| GenerateCode.js   | Hier wird gezeigt, wie man die <a href="#">Codegenerierung</a> über JScript aufruft                                                 |
| Start.js          | <a href="#">Startet</a> das als Automation Server registrierte MapForce oder stellt eine Verbindung zu einer laufenden Instanz her. |
| Readme.txt        | Diese Datei.                                                                                                                        |

### 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, a the main COM
object of an already
// running application might be returned.
try
{
 objMapForce = WScript.GetObject("", "MapForce.Application");
}
catch(err)
{
 Exit("Can't access or create MapForce.Application");
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objMapForce.Visible = true;

WScript.Echo(objMapForce.Edition + " has successfully started. ");

objMapForce.Visible = false; // will shutdown application if it has no more
COM connections
//objMapForce.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<br>Einstellungen/<Benutzername>/Eigene Dateien/<br>Altova/MapForce2012/MapForceExamples/API/ |
|------------|---------------------------------------------------------------------------------------------------------------|

|                          |                                                                             |
|--------------------------|-----------------------------------------------------------------------------|
| Windows Vista, Windows 7 | C:/Users/<Benutzername>/Documents/Altova/MapForce2012/MapForceExamples/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.

#### 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
{
 objMapForce = WScript.GetObject("", "MapForce.Application");
}
catch(err)
{
 Exit("Can't access or create MapForce.Application");
}

// if newly started, the application will start without its UI visible. Set it
// to visible.
objMapForce.Visible = true;

// ***** code snippet for "Simple Document Access"
// *****

// 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\MapForce2012\MapForceExamples\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
objMapForce.Documents.OpenDocument(strExampleFolder +
"Altova_Hierarchical_DB.mfd");

// ***** code snippet for "Simple Document Access"
// *****

// ***** code snippet for "Iteration"
// *****

// go through all open documents using a JScript Enumerator
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
 objName = iterDocs.item().Name;
 WScript.Echo("Document name: " + objName);
}

// go through all open documents using index-based access to the document
// collection
```

```
for (i = objMapForce.Documents.Count; i > 0; i--)
 objMapForce.Documents.Item(i).Close();

// ***** code snippet for "Iteration"
// *****

//objMapForce.Visible = false; // will shutdown application if it has no
// more COM connections
objMapForce.Visible = true; // will keep application running with UI visible
```

### 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<br>Einstellungen/<Benutzername>/Eigene Dateien/<br>Altova/MapForce2012/MapForceExamples/API/ |
| Windows Vista, Windows<br>7 | C:/Users/<Benutzername>/Documents/<br>Altova/MapForce2012/MapForceExamples/API/                               |

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Code generieren

Im unten stehenden JScript-Code wird gezeigt, wie man Dokumente öffnet, ein Dokument zum aktiven macht, durch die offenen Dokumente iteriert und C++-Code generiert.

### 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
{
 objMapForce = WScript.GetObject("", "MapForce.Application");
}
catch(err)
{
 Exit("Can't access or create MapForce.Application");
}

// if newly started, the application will start without its UI visible. Set it
// to visible.
objMapForce.Visible = true;

// ***** code snippet for "Simple Document Access"
// *****

// 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\MapForce2012\MapForceExamples\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
//objMapForce.Documents.OpenDocument(strExampleFolder +
"Altova_Hierarchical_DB.mfd");
objMapForce.Documents.NewDocument();

// ***** code snippet for "Simple Document Access"

// ***** code snippet for "Iteration"

objText = "";
// go through all open documents using a JScript Enumerator and generate c++
code
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
 objText += "Generated c++ code result for document " + iterDocs.item().Name
+ " :\n";
 objErrorMarkers = iterDocs.item().generateCodeEx(1); //
ENUMProgrammingLanguage.eCpp = 1

 bSuccess = true;
 for (var iterErrorMarkers = new Enumerator(objErrorMarkers);
!iterErrorMarkers.atEnd(); iterErrorMarkers.moveNext())
 {
 bSuccess = false;
 objText += "\t" + iterErrorMarkers.item().Text + "\n";
 }

 if (bSuccess)
 objText += "\tCode generation completed successfully.\n";

 objText += "\n";
}

WScript.Echo(objText);

// go through all open documents using index-based access to the document
collection
for (i = objMapForce.Documents.Count; i > 0; i--)
 objMapForce.Documents.Item(i).Close();

// ***** code snippet for "Iteration"

//objMapForce.Visible = false; // will shutdown application if it has no
more COM connections
objMapForce.Visible = true; // will keep application running with UI visible

```

### Ausführen des Skripts

Der unten aufgelistete JScript-Code steht in der Datei GenerateCode.js im Ordner JScript des Ordners API Examples zur Verfügung:

|            |                                                                                                               |
|------------|---------------------------------------------------------------------------------------------------------------|
| Windows XP | C:/Dokumente und<br>Einstellungen/<Benutzername>/Eigene Dateien/<br>Altova/MapForce2012/MapForceExamples/API/ |
|------------|---------------------------------------------------------------------------------------------------------------|

|                             |                                                                                 |
|-----------------------------|---------------------------------------------------------------------------------|
| Windows Vista, Windows<br>7 | C:/Users/<Benutzername>/Documents/<br>Altova/MapForce2012/MapForceExamples/API/ |
|-----------------------------|---------------------------------------------------------------------------------|

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

## 32.2 Objektreferenz

### Objekthierarchie

- [Application](#)
  - [Options](#)
  - [Project](#)
    - [ProjectItem](#)
  - [Documents](#)
    - [Document](#)
      - [MapForceView](#)
      - [Mapping](#)
        - [Component](#)
          - [Datapoint](#)
        - [Components](#)
        - [Connection](#)
      - [Mappings](#)
    - [ErrorMarkers](#)
      - [ErrorMarker](#)
  - [AppOutputLines](#)
    - [AppOutputLine](#)
      - AppOutputLines
      - ...
      - [AppOutputLineSymbol](#)
- [Enumerations](#)

### Beschreibung

Dieser Abschnitt enthält eine Referenzliste der MapForce API 3.0 Typbibliothek.

### 32.2.1 Applikation

Die Applikation-Schnittstelle ist die Schnittstelle zu einem MapForce application-Objekt. Sie bildet den Hauptzugriffspunkt für die MapForce-Applikation selbst. Diese Schnittstelle ist der Ausgangspunkt für alle weiteren Operationen mit MapForce oder für das Abrufen oder Erstellen anderer Automation-Objekte im Zusammenhang mit MapForce.

#### Events

[Events](#)

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

[Options](#)

[Project](#)

[Documents](#)

Applikationsstatus:

[Visible](#)

[Name](#)

[Quit](#)

[Status](#)

[WindowHandle](#)

MapForce Designs:

[NewDocument](#)

[OpenDocument](#)

[OpenURL](#)

[ActiveDocument](#)

MapForce Projekte:

[NewProject](#) (Enterprise oder Professional Edition erforderlich)

[OpenProject](#) (Enterprise oder Professional Edition erforderlich)

[ActiveProject](#) (Enterprise oder Professional Edition erforderlich)

MapForce Codegenerierung:

[HighLightSerializedMarker](#)

Globale Ressourcen:

[GlobalResourceConfig](#)

[GlobalResourceFile](#)

Versionsinformationen:

[Edition](#)

[IsAPISupported](#)

[MajorVersion](#)

[MinorVersion](#)

#### Beispiele

Die folgenden Beispiele zeigen, wie das Automation Interface von MapForce von verschiedenen Programmierungsumgebungen aus in verschiedenen Sprachen aufgerufen werden kann.

```
' ----- begin VBA example -----
' create a new instance of <SPY-MAP>.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
' ----- end example -----
```

```

' ----- begin VBScript example -----
' access a running, or create a new instance of MapForce.
' works with scripts running in the Windows scripting host.
Set objMapForce = GetObject("MapForce.Application");
' ----- end example -----

// ----- begin JScript example -----
// Access a running, or create a new instance of <MapForce
// works with scripts executed in the Windows scripting host
try
{
 objMapForce = WScript.GetObject ("", "MapForce.Application");
 // unhide application if it is a new instance
 objMapForce.Visible = true;
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }
// ----- end example -----

```

## Events

Dieses Objekt unterstützt die folgenden Events:

[OnDocumentOpened](#)

[OnProjectOpened](#)

[OnShutdown](#)

OnDocumentOpened

**Event:** [OnDocumentOpened](#) (*i\_objDocument* als [Document](#))

### Beschreibung

Das Event wird ausgelöst, wenn ein vorhandenes oder neues Dokument geöffnet wird. Das entsprechende Event zu Schließen ist [Document.OnDocumentClosed](#).

OnProjectOpened

**Event:** [OnProjectOpened](#) (*i\_objProject* als [Project](#))

### Beschreibung

Das Event wird ausgelöst, wenn ein vorhandenes oder neues Projekt in die Applikation geladen wird. Das entsprechende Event zum Schließen ist [Project.OnProjectClosed](#).

OnShutdown

**Event:** [OnShutdown](#) ()

### Beschreibung

Dieses Event wird ausgelöst, wenn die Applikation beendet wird.

## ActiveDocument

**Eigenschaft:** [ActiveDocument](#) als [Document](#) (schreibgeschützt)

### Beschreibung

Gibt das Automation-Objekt des gerade aktiven Dokuments zurück. Diese Eigenschaft gibt denselben Wert zurück wie [Documents.ActiveDocument](#).



**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ActiveProject**

**Eigenschaft:** `ActiveProject` als [Project](#) (schreibgeschützt)

**Beschreibung**

Gibt das Automation Objekt des gerade aktiven Projekts zurück.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Application**

**Eigenschaft:** `Application` als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das oberste Objekt der Applikation ab.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Documents**

**Eigenschaft:** `Documents` als [Documents](#) (schreibgeschützt)

**Beschreibung**

Gibt eine Sammlung aller gerade offenen Dokumente zurück.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Edition**

**Eigenschaft:** `Edition` als `String` (schreibgeschützt)

**Beschreibung**

Die Version (Edition) des Produkts, z.B.. Enterprise, Professional, Basic.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GlobalResourceConfig**

**Eigenschaft:** `GlobalResourceConfig` als `String`

**Beschreibung**

Ruft den Namen der aktiven Konfigurationsdatei für die globalen Ressourcen ab oder setzt ihn.

Standardmäßig trägt die Datei den Namen GlobalResources.xml.

Die Konfigurationsdatei kann umbenannt und unter einem beliebigen Pfad gespeichert werden. Sie können daher mehrere XML-Dateien für globale Ressourcen haben. Allerdings kann immer nur eine dieser Dateien pro Applikation aktiv sein und nur die in dieser Datei enthaltenen Definitionen stehen der Applikation zur Verfügung.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GlobalResourceFile**

**Eigenschaft:** `GlobalResourceFile` als String

**Beschreibung**

Ruft die Definitionsdatei für globale Ressourcen ab oder definiert sie. Standardmäßig hat die Datei den Namen GlobalResources.xml.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**HighlightSerializedMarker**

**Methode:** `HighlightSerializedMarker` (`i_strSerializedMarker` als String)

**Beschreibung**

Verwenden Sie diese Methode, um einen Pfad in einer Mapping-Datei zu markieren, die zuvor serialisiert wurde. Wenn das entsprechende Dokument noch nicht geladen ist, wird es zuerst geladen. Eine Methode zum Aufrufen eines serialisierten Markers finden Sie unter [Document.GenerateCodeEx](#).

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1007 Der in `i_strSerializedMarker` übergebene String nicht als serialisierter MapForce Marker erkannt.
- 1008 Der Marker verweist auf einen nicht mehr gültigen Pfad.

**IsAPISupported**

**Eigenschaft:** `IsAPISupported` als Boolean (schreibgeschützt)

**Beschreibung**

Gibt zurück, ob die API in dieser Version von MapForce unterstützt wird.

**Fehler**

- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**MajorVersion**

**Eigenschaft:** `MajorVersion` als Long (schreibgeschützt)

**Beschreibung**

Die Hauptversionsnummer des Produkts, z.B. 2006 für 2006 R2 SP1.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**MinorVersion**

**Eigenschaft:** [MinorVersion](#) als Long (schreibgeschützt)

**Description**

Die Zusatznummer zur Hauptversion des Produkts z.B. 2 für 2006 R2 SP1.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Name**

**Eigenschaft:** [Name](#) als String (schreibgeschützt)

**Beschreibung**

Der Name der Applikation.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**NewDocument**

**Method:** [NewDocument](#) () als [Document](#)

**Beschreibung**

Erstellt ein neues leeres Dokument. Das neu geöffnete Dokument wird das [ActiveDocument](#). Diese Methode ist die Kurzform von [Documents.NewDocument](#).

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**NewProject**

**Methode:** [NewProject](#) () als [Project](#)

**Beschreibung**

Erstellt ein neues leeres Projekt. Das aktuelle Projekt wird geschlossen. Das neue Projekt kann unter [ActiveProject](#) aufgerufen werden.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## OpenDocument

**Methode:** `OpenDocument` (*i\_strFileName* als String) als [Document](#)

### Beschreibung

Lädt eine zuvor gespeicherte Dokumentdatei und setzt die Bearbeitung fort. Das neu geöffnete Dokument wird das [ActiveDocument](#). Diese Methode ist eine Kurzform von [Documents.OpenDocument](#).

### Fehler

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## OpenProject

**Methode:** `NewProject` () als [Project](#)

### Beschreibung

Öffnet ein vorhandenes Mapforce Projekt (\*.mfp). Das aktuelle Projekt wird geschlossen. Das neu geöffnete Projekt kann unter [ActiveProject](#) aufgerufen werden.

### Fehler

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## OpenURL

**Methode:** `OpenURL` (*i\_strURL* als String, *i\_strUser* als String, *i\_strPassword* als String)

### Beschreibung

Lädt eine zuvor gespeicherte Dokumentdatei von einem URL-Pfad. Ermöglicht die Eingabe von Benutzername und Passwort.

### Fehler

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## Options

**Eigenschaft:** `Options` als [Options](#) (schreibgeschützt)

### Beschreibung

Mit dieser Eigenschaft haben Sie Zugriff auf Optionen zum Konfigurieren der Codegenerierung.

### Fehler

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## Parent

**Eigenschaft:** `Parent` als [Application](#) (schreibgeschützt)

### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1000 Das Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Quit**

**Methode:** [Quit](#) ()

**Beschreibung**

Trennt die Verbindung zu MapForce, um die Applikation beenden zu können. Der Aufruf dieser Methode ist optional, da MapForce eine Aufzeichnung aller externen COM-Verbindungen führt und getrennte Verbindungen automatisch erkennt. Nähere Informationen zum automatischen Beenden finden Sie unter der Eigenschaft [Visible](#).

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ServicePackVersion**

**Eigenschaft:** [ServicePackVersion](#) als Long (schreibgeschützt)

**Beschreibung**

Die Service Pack-Versionsnummer des Produkts, z.B. 1 für 2010 R2 SP1.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Status**

**Eigenschaft:** [Status](#) als Long (schreibgeschützt)

**Beschreibung**

Der Status der Applikation. Es ist einer der Werte der [ENUMApplicationStatus](#) Enumeration.

**Fehler**

- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Visible**

**Eigenschaft:** [Visible](#) als Boolean

**Beschreibung**

True, wenn MapForce auf dem Bildschirm angezeigt wird (unter Umständen wird es von anderen Applikationen verdeckt oder als Symbol angezeigt). False, wenn MapForce ausgeblendet ist. Der Standardwert, wenn MapForce automatisch aufgrund eines Request des Automation Server MapForce.Application gestartet wird, ist false. In allen anderen Fällen wird die Eigenschaft als true initialisiert.

Eine Applikationsinstanz, die sichtbar ist, gilt als vom Benutzer (und möglicherweise von Clients, die über die API verbunden sind) gesteuert. Sie wird nur auf einen expliziten

Benutzer-Request hin beendet. Um eine Applikationsinstanz zu beenden, setzen Sie ihre Sichtbarkeit auf false und entfernen Sie alle Referenzen auf diese Instanz aus Ihrem Programm. Die Applikationsinstanz wird automatisch beendet, wenn keine weiteren COM-Clients Referenzen darauf enthalten.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**WindowHandle**

**Eigenschaft:** `WindowHandle` () als long (schreibgeschützt)

**Beschreibung**

Ruft den Fenster-Handle der Applikation ab.

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.2 AppOutputLine

Steht für einen Meldungszeile. Im Gegensatz zu `ErrorMarker` ist die Struktur detaillierter und kann eine Sammlung von untergeordneten Zeilen enthalten, sodass eine Baumstruktur aus Meldungszeilen entsteht.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Zeilenzugriff:

[GetLineSeverity](#)

[GetLineSymbol](#)

[GetLineText](#)

[GetLineTextEx](#)

[GetLineTextWithChildren](#)

[GetLineTextWithChildrenEx](#)

Eine einzelne `AppOutputLine` besteht aus einer oder mehreren Subzeilen.

Zugriff auf Subzeilen:

[GetLineCount](#)

Eine Subzeile besteht aus einer oder mehreren Zellen.

Zugriff auf Zellen:

[GetCellCountInLine](#)

[GetCellIcon](#)

[GetCellSymbol](#)

[GetCellText](#)

[GetCellTextDecoration](#)

[GetIsCellText](#)

Unterhalb einer `AppOutputLine` können sich null, eine oder mehrere untergeordnete Zeilen befinden, die selbst vom Typ `AppOutputLine` und somit eine hierarchische Struktur bilden.

Zugriff auf untergeordnete Zeilen:

[ChildLines](#)

#### Application

**Eigenschaft:** `Application` als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1000 Das Objekt ist nicht mehr gültig.

1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### ChildLines

**Eigenschaft:** `ChildLines` als [AppOutputLines](#) (schreibgeschützt)

#### Beschreibung

Gibt eine Sammlung der Zeilen zurück, die der aktuellen Zeile direkt untergeordnet sind.

**Fehler**

- 4100 Das application-Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetCellCountInLine**

**Methode:** [GetCellCountInLine](#) (*nLine* als Long) als Long

**Beschreibung**

Ruft die Anzahl der Zellen in der Subzeile ab, die durch *nLine* in der aktuellen AppOutputLine angegeben ist.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetCellIcon**

**Methode:** [GetCellIcon](#) (*nLine* als Long, *nCell* als Long) als Long

**Beschreibung**

Ruft das Symbol der Zelle ab, die in der durch *nLine* bezeichneten Subzeile der aktuellen AppOutputLine durch *nCell* angegeben ist.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetCellSymbol**

**Methode:** [GetCellSymbol](#) (*nLine* als Long, *nCell* als Long) als [AppOutputLineSymbol](#)

**Beschreibung**

Ruft das Symbol der Zelle ab, das in der durch *nLine* bezeichneten Subzeile der aktuellen AppOutputLine durch *nCell* angegeben ist.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetCellText**

**Methode:** [GetCellText](#) (*nLine* als Long, *nCell* als Long) als String

**Beschreibung**

Ruft den Text der Zelle ab, die in der durch *nLine* bezeichneten Subzeile der aktuellen AppOutputLine durch *nCell* bezeichnet ist.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.



### GetCellTextDecoration

**Methode:** [GetCellTextDecoration](#) (*nLine* als Long, *nCell* als Long) als Long

#### Beschreibung

Ruft die Verzierung der Textzelle ab, die in der durch *nLine* bezeichneten Subzeile der aktuellen AppOutputLine durch *nCell* bezeichnet ist.

Der Wert kann einer der [ENUMAppOutputLine\\_TextDecoration](#) Werte sein.

#### Fehler

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GetIsCellText

**Methode:** [GetIsCellText](#) (*nLine* als Long, *nCell* als Long) als Boolean

#### Beschreibung

Gibt true zurück, wenn die Zelle, die in der durch *nLine* bezeichneten Subzeile der aktuellen AppOutputLine durch *nCell* bezeichnet ist, eine Textzelle ist.

#### Fehler

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GetLineCount

**Methode:** [GetLineCount](#) () als Long

#### Beschreibung

Ruft die Anzahl der Subzeilen ab, aus denen die aktuelle Zeile besteht.

#### Fehler

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GetLineSeverity

**Methode:** [GetLineSeverity](#) () als Long

#### Beschreibung

Ruft die Wichtigkeit der Zeile ab. Der Wert kann einer der [ENUMAppOutputLine\\_Severity](#) Werte sein:

#### Fehler

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GetLineSymbol

**Methode:** [GetLineSymbol](#) () als [AppOutputLineSymbol](#)

#### Beschreibung

Ruft das Symbol ab, das der gesamten Zeile zugewiesen ist.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetLineText**

**Methode:** `GetLineText` () als String

**Beschreibung**

Ruft den Inhalt der Zeile als Text ab.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetLineTextEx**

**Methode:** `GetLineTextEx` (*psTextPartSeperator* als String, *psLineSeperator* als String) als String

**Beschreibung**

Ruft unter Verwendung der angegebenen Part- und Zeilentrennzeichen den Inhalt der Zeile als Text ab.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetLineTextWithChildren**

**Methode:** `GetLineTextWithChildren` () als String

**Beschreibung**

Ruft den Inhalt der Zeile inklusive aller untergeordneten Zeilen als Text ab.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GetLineTextWithChildrenEx**

**Methode:** `GetLineTextWithChildrenEx` (*psPartSep* als String, *psLineSep* als String, *psTabSep* als String, *psItemSep* als String) als String

**Beschreibung**

Ruft den Inhalt der Zeile inklusive aller untergeordneten Zeilen als Text ab, wobei die angegebenen Trennzeichen für Teile, Tabulatoren und Elemente verwendet werden.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** **Parent** als [AppOutputLines](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt entsprechend dem Objektmodell.

**Fehler**

- 4100 Das Objekt ist nicht mehr gültig.
- 4101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.3 AppOutputLines

Steht für eine Sammlung von AppOutputLine Meldungszeilen.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Iterieren durch die Sammlung:

[Count](#)

[Item](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

- 4000 Das Objekt ist nicht mehr gültig.
- 4001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Count

**Eigenschaft:** [Count](#) als Integer (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Zeilen in der Sammlung ab.

#### Fehler

- 4000 Das Objekt ist nicht mehr gültig.
- 4001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Item

**Eigenschaft:** [Item](#) ([nIndex](#) als Integer) als [AppOutputLine](#) (schreibgeschützt)

#### Beschreibung

Ruft die Zeile am Index nIndex aus der Sammlung ab. Indizes beginnen mit 1.

#### Fehler

- 4000 Das Objekt ist nicht mehr gültig.
- 4001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Parent

**Eigenschaft:** [Parent](#) als [AppOutputLine](#) (schreibgeschützt)

#### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

#### Fehler

- 4000 Das Objekt ist nicht mehr gültig.
- 4001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.4 AppOutputLineSymbol

Ein AppOutputLineSymbol stellt einen Link in einer AppOutputLine Meldungszeile dar, der im MapForce Meldungsfenster angeklickt werden kann.

Wird auf eine Zelle einer AppOutputLine oder auf die gesamte Zeile angewendet-

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Zugriff auf die AppOutputLineSymbol-Methoden:

[GetSymbolHREF](#)

[GetSymbolID](#)

[IsSymbolHREF](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation auf.

#### Fehler

- 4200 Das Objekt ist nicht mehr gültig.
- 4201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### GetSymbolHREF

**Methode:** [GetSymbolHREF](#) () als String

#### Beschreibung

Wenn das Symbol vom Typ URL ist, wird die URL als String zurückgegeben.

#### Fehler

- 4200 Das Objekt ist nicht mehr gültig.
- 4201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### GetSymbolID

**Methode:** [GetSymbolHREF](#) () als Long

#### Beschreibung

Ruft die ID des Symbols ab.

#### Fehler

- 4200 Das Objekt ist nicht mehr gültig.
- 4201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### IsSymbolHREF

**Methode:** [IsSymbolHREF](#) () als Boolean

#### Beschreibung

Gibt an, ob das Symbol vom Typ URL ist.

**Fehler**

- 4200 Das Objekt ist nicht mehr gültig.
- 4201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** **Parent** als [Application](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 4200 Das Objekt ist nicht mehr gültig.
- 4201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.5 Component

Eine Komponente stellt eine [MapForce-Komponente](#) dar.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Komponenteneigenschaften:

[HasIncomingConnections](#)

[HasOutgoingConnections](#)

[CanChangeInputInstanceFile](#)

[CanChangeOutputInstanceFile](#)

[ID](#)

[IsParameterInputRequired](#)

[IsParameterSequence](#)

[Name](#)

[Preview](#)

[Schema](#)

[SubType](#)

[Type](#)

Eigenschaften im Zusammenhang mit Instanzen:

[InputInstanceFile](#)

[OutputInstanceFile](#)

Datapoints:

[GetRootDatapoint](#)

Ausführung:

[GenerateOutput](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1200 Das Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### CanChangeInputInstanceFile

**Eigenschaft:** [CanChangeInputInstanceFile](#) als Boolean (schreibgeschützt)

#### Beschreibung

Gibt an, ob der Name der Input-Instanzdatei geändert werden kann.

Gibt "false" zurück, wenn die Komponente einen Dateinamen-Node hat und dieser Node eine Verbindung auf seiner linken (Input-) Seite hat. Andernfalls wird "true" zurückgegeben.

Wenn die Komponente keinen Dateinamen-Node hat, wird "false" zurückgegeben.

#### Fehler



- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### CanChangeOutputInstanceFile

**Eigenschaft:** CanChangeOutputInstanceFile as Boolean (schreibgeschützt)

#### Beschreibung

Gibt an, ob der Name der Output-Instanzdatei geändert werden kann.

Gibt "false" zurück, wenn die Komponente einen Dateinamen-Node hat und dieser Node eine Verbindung auf seiner linken (Input-) Seite hat. Andernfalls wird "true" zurückgegeben. Wenn die Komponente keinen Dateinamen-Node hat, wird "false" zurückgegeben.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GenerateOutput

**Methode:** [GenerateOutput](#) ([out] *pbError* als Boolean) als [AppOutputLines](#)

#### Beschreibung

Generiert mit Hilfe einer MapForce-internen Mapping-Sprache die Ausgabedatei(en), die im Mapping nur für die aktuelle Komponente definiert sind. Der Name/Die Namen der Ausgabedatei(en) sind als Eigenschaft der aktuellen Komponente definiert, welche im Mapping für diese Generierung das Ausgabeelement ist.

#### Anmerkungen

*pbError* ist ein Parameter, der nur für die Ausgabe verwendet wird. Sie erhalten nur dann einen Wert, wenn die aufrufende Sprache Ausgabeparameter unterstützt. Falls nicht, bleibt der hier übergebene Wert unverändert, wenn die Funktion fertig ist.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GetRootDatapoint

**Methode:** [GetRootDatapoint](#)( *side* als [ENUMComponentDatapointSide](#), *strNamespace* als String, *strLocalName* als String, *strParameterName* als String ) als [Datapoint](#)

#### Beschreibung

Ruft einen Root Datapoint auf der linken (Input) oder rechten (Output) Seite einer Komponente ab. Um Subelemente und untergeordnete Elemente davon aufzurufen, stellt das Datapoint-Objekt weitere Methoden bereit.

Der Parameter *side* gibt an, ob ein Input oder Output Datapoint einer Komponente abgerufen werden soll.

Der angegebene Namespace und der lokale Name geben den spezifischen Namen des Node an, dessen Datapoint abgerufen werden soll. Bei Komponenten mit Strukturinformationen wie z. B. Schemakomponenten muss der Namespace zusammen mit dem lokalen Namen angegeben

werden, oder Sie übergeben einfach einen leeren String für den Namespace.

Dateibasierte Komponenten wie die Schemakomponente enthalten einen speziellen Node an ihrer Root, den Dateinamen-Node. Hier findet `GetRootDatapoint` nur den Dateinamen-Node. Sie müssen den Namespace "`http://www.altova.com/mapforce`" und lokalen Namen "`FileInstance`" übergeben, um einen Datapoint dieses Node abzurufen.

Der angegebene Parameter sollte ein leerer String sein, es sei denn, es handelt sich bei der betreffenden Komponente um eine Funktionsaufruf-Komponente. Da eine benutzerdefinierte Funktion Input- oder Output-Parameter derselben Struktur enthalten könnte, kann eine Funktionsaufruf-Komponente, die diese benutzerdefinierte Funktion aufruft, mehr als einen Root-Node mit einem identischen Namespace und lokalen Namen haben.

Diese unterscheiden sich dann nur in ihren Parameternamen, bei denen es sich eigentlich um die Namen der entsprechenden Parameterkomponenten in Mapping der benutzerdefinierten Funktion selbst handelt.

Der Parametername muss nicht zwingend angegeben werden. In diesem Fall gibt die Methode den ersten Root Datapoint zurück, der mit dem angegebenen Namespace und lokalen Namen übereinstimmt.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1231 Datapoint wurde nicht gefunden.

**HasIncomingConnections**

**Eigenschaft:** `HasIncomingConnections` als Boolean (schreibgeschützt)

**Beschreibung**

Gibt an, ob die Komponente eingehende Verbindungen hat (auf der linken Seite). Der Dateinamen-Node ist dadurch nicht betroffen. Eine eingehende Verbindung am Dateinamen-Node hat keine Auswirkung auf den zurückgegebenen Wert.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**HasOutgoingConnections**

**Eigenschaft:** `HasOutgoingConnections` als Boolean (schreibgeschützt)

**Beschreibung**

Gibt an, ob die Komponente ausgehende Verbindungen hat (auf der rechten Seite).

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ID**

**Eigenschaft:** `ID` als Unsigned Long (schreibgeschützt)

**Beschreibung**

Ruft die ID der Komponente ab.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**InputInstanceFile**

**Eigenschaft:** [InputInstanceFile](#) als String

**Beschreibung**

Ruft die Input-Instanzdatei der Komponente ab oder definiert sie.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**IsParameterInputRequired**

**Eigenschaft:** [IsParameterInputRequired](#) als Boolean

**Beschreibung**

Ruft ab oder definiert, ob für die Input-Parameterkomponente in der Funktionsaufruf-Komponente der benutzerdefinierten Funktion, in der sich diese Input-Parameterkomponente befindet, eine eingehende Verbindung erforderlich ist. Diese Eigenschaft funktioniert nur bei Komponenten, die Input-Parameterkomponenten sind.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1233 Diese Operation funktioniert nur bei einer Input- oder Output-Parameterkomponente
- 1240 Eine Änderung des Dokuments ist nicht zulässig. Es ist schreibgeschützt.

**IsParameterSequence**

**Eigenschaft:** [IsParameterSequence](#) als Boolean

**Beschreibung**

Ruft ab oder definiert, ob die Input- oder Output-Parameterkomponente Sequenzen unterstützt. Diese Eigenschaft funktioniert nur bei Komponenten, die Input-Parameterkomponenten sind.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1233 Diese Operation funktioniert nur bei einer Input- oder Output-Parameterkomponente
- 1240 Eine Änderung des Dokuments ist nicht zulässig. Es ist schreibgeschützt.

**Name**

**Eigenschaft:** [Name](#) als String

**Beschreibung**

Ruft den Namen der Komponente ab oder definiert ihn.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1246 Die Komponente unterstützt die Definition des Namens nicht.
- 1247 Ungültiger Komponentename

**OutputInstanceFile**

**Eigenschaft:** `OutputInstanceFile` als String

**Beschreibung**

Ruft die Output-Instanzdatei der Komponente ab oder definiert sie.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** `Parent` als [Mapping](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Preview**

**Eigenschaft:** `Preview` als Boolean

**Beschreibung**

Ruft ab oder definiert, ob die Komponente die aktuelle Vorschaukomponente ist.

Diese Eigenschaft funktioniert nur bei Komponenten, bei denen es sich um Zielkomponenten im Hauptmapping des Dokuments handelt. Es kann im Hauptmapping immer nur eine Zielkomponente gleichzeitig als Vorschaukomponente verwendet werden.

Sie können diese Eigenschaft auf "true" setzen. Damit wird dann die Eigenschaft "Preview" bei allen anderen Komponenten implizit auf "false" gesetzt.

Wenn es im Hauptmapping nur eine einzige Zielkomponente gibt, so ist diese auch die Vorschaukomponente.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1234 Nur eine Zielkomponente im Hauptmapping kann als Vorschaukomponente definiert werden.
- 1235 Eine Komponente kann nicht als Nicht-Vorschaukomponente definiert werden. Definieren Sie statt dessen eine andere Komponente als Vorschaukomponente.

**Schema**

**Eigenschaft:** [Schema](#) als String (schreibgeschützt)

**Beschreibung**

Ruft den Schemadateinamen der Komponente ab.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**SubType**

**Eigenschaft:** [SubType](#) als [ENUMComponentSubType](#) (schreibgeschützt)

**Beschreibung**

Ruft den Subtyp der Komponente ab.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Type**

**Eigenschaft:** [Type](#) als [ENUMComponentType](#) (schreibgeschützt)

**Beschreibung**

Ruft den Typ der Komponente ab.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**UsageKind**

**Eigenschaft:** [UsageKind](#) als [ENUMUsageKind](#) (read-only)

**Beschreibung**

Ruft die Verwendungsart der Komponente ab.

**Errors**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.6 Components

Stellt eine Sammlung von component-Objekten dar.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Iterieren durch die Sammlung:

[Count](#)

[Item](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1200 Das Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Count

**Eigenschaft:** [Count](#) als Integer (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Komponenten in der Sammlung ab.

#### Fehler

1200 Das Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Item

**Eigenschaft:** [Item](#) ([nIndex](#) als Integer) als [Component](#) (schreibgeschützt)

#### Beschreibung

Ruft die Komponente am Index [nIndex](#) aus der Sammlung ab. Indizes beginnen mit 1.

#### Fehler

1200 Das Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Parent

**Eigenschaft:** [Parent](#) als [Mapping](#) (schreibgeschützt)

#### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.7 Connection

Ein Connection-Objekt steht für einen Konnektor zwischen zwei Komponenten.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Eigenschaften

[ConnectionType](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das Objekt auf oberster Ebene einer Applikation ab.

#### Fehler

- 2100 Das Objekt ist nicht mehr gültig.
- 2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### ConnectionType

**Eigenschaft:** [ConnectionType](#) als [ENUMConnectionType](#)

#### Beschreibung

Ruft den Verbindungstyp ab oder definiert ihn.

#### Fehler

- 2100 Das Objekt ist nicht mehr gültig.
- 2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2102 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 2103 Der Verbindungstyp konnte nicht geändert werden.

#### Parent

**Eigenschaft:** [Parent](#) als [Mapping](#) (schreibgeschützt)

#### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

#### Fehler

- 2100 Das Objekt ist nicht mehr gültig.
- 2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.



### 32.2.8 Datapoint

Ein Datapoint-Objekt repräsentiert ein Input- oder Output-Symbol einer Komponente.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Methoden

[GetChild](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das Objekt auf der obersten Ebene der Applikation ab.

#### Fehler

- 2000 Das Objekt ist nicht mehr gültig.
- 2001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### GetChild

**Methode:** `GetChild( strNamespace as String, strLocalName as String, searchFlags als ENUMSearchDatapointFlags )` als [Datapoint](#)

#### Beschreibung

Sucht nach einem direkten Child Datapoint des aktuellen Datapoint. Sucht nach Namespace und lokalem Namen.

Such-Flags können als Kombination von Werten der `ENUMSearchDatapointFlags` Enumeration übergeben werden (kombiniert mittels binärem OR).

Eine Schemakomponente mit Elementen, die mixed Content enthalten, wobei jeder einen zusätzlichen Child Node, den so genannten `text()` Node enthält. Um einen Datapoint eines `text()` Node abzurufen, müssen Sie in `strNamespace` einen leeren String sowie in `strLocalName` und `eSearchDatapointElement` in `searchFlags` `"#text"` übergeben.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2002 Datapoint wurde nicht gefunden.

#### Parent

**Eigenschaft:** [Parent](#) als [Component](#) (schreibgeschützt)

#### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

#### Fehler

- 2000 Das Objekt ist nicht mehr gültig.
- 2001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.9 Document

Ein Document-Objekt repräsentiert ein MapForce-Dokument (eine geladene MDF-Datei). Ein Dokument enthält ein Hauptmapping und null oder mehr lokale benutzerdefinierte Funktionsmappings.

#### Events

[Events](#)

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Dateibehandlung:

[Activate](#)

[Close](#)

[FullName](#)

[Name](#)

[Path](#)

[Saved](#)

[Save](#)

[SaveAs](#)

Mapping-Behandlung:

[MainMapping](#)

[Mappings](#)

[CreateUserDefinedFunction](#)

Komponentenbehandlung:

[FindComponentByID](#)

Codegenerierung:

[OutputSettings](#) [ApplicationName](#)

[JavaSettings](#) [BasePackageName](#)

[GenerateCHashCode](#)

[GenerateCodeEx](#)

[GenerateCppCode](#)

[GenerateJavaCode](#)

[GenerateXQuery](#)

[GenerateXSLT](#)

[GenerateXSLT2](#)

[HighlightSerializedMarker](#)

Mapping-Ausführung:

[GenerateOutput](#)

[GenerateOutputEx](#)

Aufruf der Ansicht:

[MapForceView](#)

Nicht mehr verwendet:

[OutputSettings](#) [Encoding](#)

## Events

Dieses Objekt unterstützt die folgenden Events:

[OnDocumentClosed](#)

[OnModifiedFlagChanged](#)

OnDocumentClosed

**Event:** [OnDocumentClosed](#) (*i\_objDocument* als [Document](#))

### Beschreibung

Dieses Event wird ausgelöst, wenn ein Dokument geschlossen wird. Das an den Event Handler übergebene Dokumentobjekt sollte nicht aufgerufen werden. Das entsprechenden Event zum Öffnen ist [Application.OnDocumentOpened](#).

OnModifiedFlagChanged

**Event:** [OnModifiedFlagChanged](#) (*i\_bIsModified* als Boolean)

### Beschreibung

Dieses Event wird ausgelöst, wenn sich der Änderungsstatus eines Dokuments ändert.

## Activate

**Methode:** [Activate](#) ()

### Beschreibung

Macht das Dokument zum aktiven Dokument.

### Fehler

1200 Das application-Objekt ist nicht mehr gültig.

## Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

### Beschreibung

Ruft das oberste Objekt der Applikation ab.

### Fehler

1200 Das application-Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## Close

**Methode:** [Close](#) ()

### Beschreibung

Schließt das Dokument ohne es zu speichern.

### Fehler

1200 Das application-Objekt ist nicht mehr gültig.

1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### CreateUserDefinedFunction

**Methode:** `CreateUserDefinedFunction`( `strFunctionName` als String, `strLibraryName` als String, `strSyntax` als String, `strDetails` als String, `bInlinedUse` als Boolean ) als [Mapping](#)

#### Beschreibung

Erstellt eine benutzerdefinierte Funktion im aktuellen Dokument.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1208 Benutzerdefinierte Funktion konnte nicht erstellt werden.
- 1209 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.

### FindComponentByID

**Methode:** `FindComponentByID` (`nID` als Unsigned Long) als [Component](#)

#### Beschreibung

Durchsucht das gesamte Dokument und alle seine Mappings nach der Komponente mit der angegebenen ID.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### FullName

**Eigenschaft:** `FullName` als String

#### Beschreibung

Pfad und Name der Dokumentdatei.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### GenerateCHashCode

**Methode:** `GenerateCHashCode` ()

#### Beschreibung

Generiert C#-Code, der das Mapping ausführt. Verwendet zum Konfigurieren der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1205 Fehler bei der Codegenerierung.

#### Siehe auch

Codegenerierung

## GenerateCodeEx

**Methode:** [GenerateCodeEx](#) (*i\_nLanguage* as [ENUMProgrammingLanguage](#)) as [ErrorMarkers](#)

### Beschreibung

Generiert C++-Code, der das Mapping ausführt. Der Parameter *i\_nLanguage* definiert die Zielsprache. Die Methode gibt ein Objekt zurück, das verwendet werden kann, um alle vom Codegenerator erstellten Meldungen aufzuzählen. Dabei handelt es sich um dieselben Meldungen, die im Meldungsfenster von MapForce angezeigt werden.

### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1205 Fehler bei der Codegenerierung.

### Siehe auch

Codegenerierung

## GenerateCppCode

**Methode:** [GenerateCppCode](#) ()

### Beschreibung

Generiert C++-Code, der das Mapping ausführt. Verwendet zum Konfigurieren der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1205 Fehler bei der Codegenerierung.

### Siehe auch

Codegenerierung

## GenerateJavaCode

**Methode:** [GenerateJavaCode](#) ()

### Beschreibung

Generiert Java-Code, der das Mapping ausführt. Verwendet zur Konfigurierung der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1205 Fehler bei der Codegenerierung.

### Siehe auch

Codegenerierung

**GenerateOutput**

**Methode:** [GenerateOutput \(\)](#)

**Beschreibung**

Generiert alle im Mapping definierten Ausgabedateien mittels einer MapForce-internen Mapping-Sprache. Die Namen der Ausgabedateien werden als Eigenschaften der Ausgabeelemente im Mapping definiert.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1206 Fehler bei der Ausführung eines Mapping-Algorithmus.

**Siehe auch**

Codegenerierung

**GenerateOutputEx**

**Methode:** [GenerateOutputEx \(\)](#) als [AppOutputLines](#)

**Beschreibung**

Generiert alle im Mapping definierten Ausgabedateien mittels einer MapForce-internen Mapping-Sprache. Die Namen der Ausgabedateien werden als Eigenschaften der Ausgabeelemente im Mapping definiert. Diese Methode ist mit Ausnahme des Rückgabewerts, der die erzeugten Meldungen, Warnmeldungen und Fehler in Form einer Baumstruktur von [AppOutputLine](#)s enthält, identisch mit [GenerateOutput](#).

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1206 Fehler bei der Ausführung eines Mapping-Algorithmus.

**Siehe auch**

Codegenerierung

**GenerateXQuery**

**Methode:** [GenerateXQuery \(\)](#)

**Beschreibung**

Generiert Mappingcode als XQuery. Verwendet zur Konfigurierung der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1204 Error during XSLT/XSLT2/XQuery code generation.

**Siehe auch**

Codegenerierung

**GenerateXSLT**

**Methode:** [GenerateXSLT \(\)](#)

**Beschreibung**

Generiert Mappingcode als XSLT. Verwendet zur Konfigurierung der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1204 Error during XSLT/XSLT2/XQuery code generation.

**Siehe auch**

Codegenerierung

**GenerateXSLT2**

**Methode:** [GenerateXSLT2](#) ()

**Beschreibung**

Generiert Mappingcode als XSLT2. Verwendet zur Konfigurierung der Codegenerierung die in [Application.Options](#) definierten Eigenschaften.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1204 Error during XSLT/XSLT2/XQuery code generation.

**Siehe auch**

Codegenerierung

**HighlightSerializedMarker**

**Methode:** [HighlightSerializedMarker](#) (*i\_strSerializedMarker* als String)

**Beschreibung**

Verwenden Sie diese Methode, um einen Pfad in einer Mapping-Datei zu markieren, die zuvor serialisiert wurde. Wenn das entsprechende Dokument noch nicht geladen ist, wird es zuerst geladen. Eine Methode zum Aufrufen eines serialisierten Markers finden Sie unter [GenerateCodeEx](#).

**Fehler**

- 1000 Das application-Objekt ist nicht mehr gültig.
- 1001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1007 Der in *i\_strSerializedMarker* übergebene String wird nicht als serialisierter MapForce Marker erkannt.
- 1008 Der Marker verweist auf einen nicht mehr gültigen Pfad.

**JavaSettings\_BasePackageName**

**Eigenschaft:** [JavaSettings\\_BasePackageName](#) als String

**Beschreibung**

Setzt den beim Generieren von Java-Code verwendeten Base Package-Namen oder ruft ihn ab. Diese Eigenschaft steht im UI-Dialogfeld für die Dokumenteinstellungen zur Verfügung.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

### MainMapping

**Eigenschaft:** [MainMapping](#) als [Mapping](#) (schreibgeschützt)

#### Beschreibung

Ruft das Hauptmapping des Dokuments auf.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### MapForceView

**Eigenschaft:** [MapForceView](#) als [MapForceView](#) (schreibgeschützt)

#### Beschreibung

Mit dieser Eigenschaft erhalten Sie Zugriff auf die Funktionalitäten der MapForce Ansicht.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### Mappings

**Eigenschaft:** [Mappings](#) als [Mappings](#) (schreibgeschützt)

#### Beschreibung

Gibt eine Sammlung der im Dokument enthaltenen Mappings zurück.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### Name

**Eigenschaft:** [Name](#) als String

#### Beschreibung

Name der Dokumentdatei ohne Dateipfad.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### OutputSettings\_ApplicationName

**Eigenschaft:** [OutputSettings\\_ApplicationName](#) als String

#### Beschreibung

Setzt den im Dialogfeld "Dokumenteinstellungen" verfügbaren Applikationsnamen bzw. ruft diesen ab.



**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**OutputSettings\_Encoding (nicht mehr verwendet)**

**Eigenschaft:** [OutputSettings\\_Encoding](#) als String

**Beschreibung**

*wird nicht mehr verwendet*

Diese Eigenschaft wird nicht mehr unterstützt. Es gibt keine Kodierungseinstellungen mehr für die Mapping-Ausgabe. Die Kodierungseinstellungen werden für die einzelnen Komponenten festgelegt.

**Siehe auch**

Codegenerierung

**Parent**

**Eigenschaft:** [Parent](#) als [Documents](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Path**

**Eigenschaft:** [Path](#) als String

**Beschreibung**

Pfad der Dokumentdatei ohne Namen.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Save**

**Methode:** [Save](#) ()

**Beschreibung**

Speichert das Dokument in der durch [Document.FullName](#) definierten Datei.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**SaveAs**

**Methode:** `SaveAs` (*`i_strFileName`* als String)

**Beschreibung**

Speichert das Dokument unter dem angegebenen Dateinamen und setzt [`Document.FullName`](#) auf diesen Wert, wenn die Operation erfolgreich war.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Saved**

**Eigenschaft:** `Saved` als Boolean (schreibgeschützt)

**Beschreibung**

True, wenn das Dokument seit dem letzten Speichern nicht geändert wurde, andernfalls false.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.10 Documents

Stellt eine Sammlung von document-Objekten dar.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Öffnen und Erstellen von Mappings:

[OpenDocument](#)

[NewDocument](#)

Iterieren durch die Sammlung:

[Count](#)

[Item](#)

[ActiveDocument](#)

#### ActiveDocument

**Eigenschaft:** [ActiveDocument](#) als [Document](#) (schreibgeschützt)

#### Beschreibung

Ruft das aktive Dokument ab. Wenn kein Dokument offen ist, wird null zurückgegeben.

#### Fehler

1600 Das Objekt ist nicht mehr gültig.

1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1600 Das Objekt ist nicht mehr gültig.

1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Count

**Eigenschaft:** [Count](#) als Integer (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Dokumente in der Sammlung ab.

#### Fehler

1600 Das Objekt ist nicht mehr gültig.

1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Item**

**Eigenschaft:** **Item** (*nIndex* als Integer) als [Document](#) (schreibgeschützt)

**Beschreibung**

Ruft das Dokument unter dem *nIndex* aus der Sammlung ab. Indizes beginnen mit 1.

**Fehler**

- 1600 Das Objekt ist nicht mehr gültig.
- 1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**NewDocument**

**Methode:** **NewDocument** () als [Document](#)

**Beschreibung**

Erstellt das neue Dokument, fügt es zum Ender der Sammlung hinzu und machte es zum aktiven Dokument.

**Fehler**

- 1600 Das Objekt ist nicht mehr gültig.
- 1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**OpenDocument**

**Methode:** **OpenDocument** (*strFilePath* als String) als [Document](#)

**Beschreibung**

Öffnet ein vorhandenes Mapping-Dokument (\* .mfd). Fügt das neu geöffnete Dokument zum Ende der Sammlung hinzu und macht es zum aktiven Dokument.

**Fehler**

- 1600 Das Objekt ist nicht mehr gültig.
- 1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** **Parent** als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das Parent-Objekt gemäß dem Objektmodell auf.

**Fehler**

- 1600 Das Objekt ist nicht mehr gültig.
- 1601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.11 ErrorMarker

Repräsentiert eine einfache Meldungszeile. Im Unterschied zu AppOutputLine haben errorMarkers keine hierarchische Struktur.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Zugriff auf Meldungsinformationen:

[DocumentFileName](#)

[ErrorLevel](#)

[Highlight](#)

[Serialization](#)

[Text](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### DocumentFileName

**Eigenschaft:** [DocumentFileName](#) als String (schreibgeschützt)

#### Beschreibung

Ruft den Namen der Mapping-Datei ab, mit der der Fehler-Marker verknüpft ist.

#### Fehler

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### ErrorLevel

**Eigenschaft:** [ErrorLevel](#) als [ENUMCodeGenErrorLevel](#) (schreibgeschützt)

#### Beschreibung

Ruft die Schwere des Fehlers ab.

#### Fehler

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Highlight

**Methode:** [Highlight\(\)](#)

#### Beschreibung

Markiert das Element, mit dem der Fehler-Marker verknüpft ist. Wenn das entsprechende Dokument nicht offen ist, wird es geöffnet.

**Fehler**

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1008 Der Marker verweist auf einen nicht mehr gültigen Pfad.

**Serialization**

**Eigenschaft:** [Serialization](#) als String (schreibgeschützt)

**Beschreibung**

Serialisiert den Fehlermarker in einen String. Verwenden Sie diesen String in Aufrufen von [Application.HighlightSerializedMarker](#) oder [Document.HighlightSerializedMarker](#) zum Hervorheben des markierten Elements im Mapping. Der String kann in anderen Instanziierungen von MapForce oder seinem Control verwendet werden.

**Fehler**

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Text**

**Eigenschaft:** [Text](#) als String (schreibgeschützt)

**Beschreibung**

Ruft den Meldungstext ab.

**Fehler**

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** [Parent](#) als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das Parent-Objekt gemäß dem Objektmodell auf.

**Fehler**

- 1900 Das Objekt ist nicht mehr gültig.
- 1901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.12 ErrorMarkers

Stellt eine Sammlung von ErrorMarker-Objekten dar.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Iterieren durch die Sammlung:

[Count](#)

[Item](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1800 Das Objekt ist nicht mehr gültig.

1801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Count

**Eigenschaft:** [Count](#) als Integer (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Fehler-Marker in der Sammlung auf.

#### Fehler

1800 Das Objekt ist nicht mehr gültig.

1801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Item

**Eigenschaft:** [Item](#) ([nIndex](#) als Integer) als [ErrorMarker](#) (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Fehlermarker am [nIndex](#) aus der Sammlung ab. Indizes beginnen mit 1.

#### Fehler

1800 Das Objekt ist nicht mehr gültig.

1801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Parent

**Eigenschaft:** [Parent](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

1800 Das Objekt ist nicht mehr gültig.

1801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.



### 32.2.13 MapForceView

Stellt die aktuelle Ansicht eines Dokument auf dem Register "Mapping" dar.  
Ein Dokument hat genau eine MapForce-Ansicht (MapForceView), in der das aktuell aktive Mapping angezeigt wird.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Ansichtsaktivierung und Ansichtseigenschaften:

[Active](#)

[ShowItemTypes](#)

[ShowLibraryInFunctionHeader](#)

[HighlightMyConnections](#)

[HighlightMyConnectionsRecursively](#)

Mappen in Zusammenhang stehender Eigenschaften:

[ActiveMapping](#)

[ActiveMappingName](#)

Hinzufügen von Elementen:

[InsertwSDLCall](#)

[InsertXMLFile](#)

[InsertXMLSchema](#)

[InsertXMLSchemaWithSample](#)

#### Active

**Eigenschaft:** [Active](#) als Boolean

#### Beschreibung

Mit Hilfe dieser Eigenschaft können Sie abfragen, ob die Mappingansicht die aktive Ansicht ist oder Sie können diese Ansicht zur aktiven machen.

#### Fehler

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### ActiveMapping

**Property:** [ActiveMapping](#) als [Mapping](#)

#### Beschreibung

Ruft das aktuell aktive Mapping im Dokument auf, zu dem diese MapForce-Ansicht (MapForceView) gehört, ab, bzw. definiert es.

#### Fehler

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### ActiveMappingName

**Eigenschaft:** [ActiveMappingName](#) als [String](#)

**Beschreibung**

Ruft das aktuell aktive Mapping im Dokument auf, zu dem diese MapForce-Ansicht (MapForceView) gehört, nach Name ab, bzw. definiert es.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Application**

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das oberste Objekt der Applikation ab.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**HighlightMyConnections**

**Eigenschaft:** [HighlightMyConnections](#) als Boolean

**Beschreibung**

Mit dieser Eigenschaft wird definiert, ob Verbindungen vom ausgewählten Element nur markiert werden sollen.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**HighlightMyConnectionsRecursivey**

**Eigenschaft:** [HighlightMyConnectionsRecursively](#) als Boolean

**Beschreibung**

Mit dieser Eigenschaft wird definiert, ob nur die Verbindungen, die direkt von und zu dem ausgewählten Datenelement führen, markiert werden sollen.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**InsertWSDLCall**

**Methodr:** [InsertWSDLCall](#) ([i\\_strWSDLFileName](#) als String)

**Beschreibung**

Fügt eine neue WSDL Call-Komponente zum Mapping hinzu.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**InsertXMLFile (nicht mehr verwendet)**

**Methode:** `InsertXMLFile` (*`i_strXMLFileName`* als String, *`i_strRootElement`* als String)

**Beschreibung**

*Wird nicht mehr verwendet*

MapForceView.InsertXMLFile wird nicht mehr verwendet. Verwenden Sie stattdessen Mapping.InsertXMLFile.

Fügt eine neue Komponente zum Mapping hinzu.

Die interne Struktur der Komponente hängt vom Schema ab, das in der angegebenen XML-Datei definiert ist.

Der zweite Parameter definiert das Root-Element dieses Schemas, wenn mehr als eines zur Auswahl steht.

Bei Übergabe eines leeren String als Root-Element wird das Root-Element der XML-Datei verwendet. Wenn mehrere Elemente zur Auswahl stehen, wird - unabhängig davon, ob MapFoce angezeigt wird oder nicht, ein Dialogfeld "Root-Element auswählen" angezeigt.

Die angegebene XML-Datei wird als Beispiel-Input zum Auswerten des Mappings verwendet.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**InsertXMLSchema (nicht mehr verwendet)**

**Methode:** `InsertXMLSchema` (*`i_strSchemaFileName`* als String, *`i_strRootElement`* als String)

**Beschreibung**

*Wird nicht mehr verwendet.*

MapForceView.InsertXMLSchema wird nicht mehr verwendet. Verwenden Sie stattdessen Mapping.InsertXMLSchema.

Fügt eine neue Komponente zum Mapping hinzu.

Die interne Struktur der Komponente hängt von der angegebenen Schemadatei ab.

Der zweite Parameter definiert das Root-Element dieses Schemas, wenn mehr als eines zur Auswahl steht. Wenn das übergebene Root-Element ein leerer String ist und mehrere Elemente zur Auswahl stehen, wird - unabhängig davon, ob MapFoce angezeigt wird oder nicht, ein Dialogfeld "Root-Element auswählen" angezeigt.

Diesem Element ist keine XML-Input-Beispieldatei zugeordnet.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**InsertXMLSchemaWithSample (nicht mehr verwendet)**

**Methode:** `InsertXMLSchemaWithSample` (*`i_strSchemaFileName`* als String, *`i_strXMLSampleName`* als String, *`i_strRootElement`* als String)

**Beschreibung**

*Wird nicht mehr verwendet.*

MapForceView.InsertXMLSchemaWithSample wird nicht mehr verwendet. Verwenden Sie stattdessen Mapping.InsertXMLFile. Beachten Sie, dass zur Übergabe des Root-Elements kein Parameter erforderlich ist. Das Root-Element wird automatisch als Root-Elementname der XML-Datei definiert.

Fügt eine neue Komponente zum Mapping hinzu.

Die interne Struktur der Komponente hängt von der angegebenen Schemadatei ab.

Der zweite Parameter wird als die XML-Input-Beispieldatei für die Auswertung des Mapping gespeichert.

Der dritte Parameter definiert das Root-Element dieses Schemas, wenn mehr als eines zur Auswahl steht.

Bei Übergabe eines leeren String als Root-Element wird das Root-Element der XML-Beispieldatei verwendet.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** `Parent` als [Document](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ShowItemTypes**

**Eigenschaft:** `ShowItemTypes` als `Boolean`

**Beschreibung**

Mit dieser Eigenschaft wird definiert, ob Elementtypen im Mapping-Diagramm angezeigt werden sollen.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ShowLibraryInFunctionHeader**

**Eigenschaft:** `ShowLibraryInFunctionHeader` als `Boolean`

**Beschreibung**

Mit dieser Eigenschaft wird definiert, ob der Name der Funktionsbibliothek Teil von Funktionsnamen sein soll.

**Fehler**

- 1300 Das application-Objekt ist nicht mehr gültig.
- 1301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.14 Mapping

Ein Mapping-Objekt repräsentiert ein Mapping in einem Dokument, also das Hauptmapping oder ein benutzerdefiniertes Funktionsmapping.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Mapping-Eigenschaften:

[IsMainMapping](#)

[Name](#)

Komponenten im Mapping:

[Components](#)

Hinzufügen von Datenelementen:

[CreateConnection](#)

[InsertFunctionCall](#)

[InsertXMLFile](#)

[InsertXMLSchema](#)

[InsertXMLSchemaInputParameter](#)

[InsertXMLSchemaOutputParameter](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Components

**Eigenschaft:** [Components](#) als [Components](#) (schreibgeschützt)

#### Beschreibung

Gibt eine Sammlung aller Komponenten im aktuellen Mapping zurück.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### CreateConnection

**Methode:** [CreateConnection](#)( [DatapointFrom](#) als [Datapoint](#), [DatapointTo](#) als [Datapoint](#) ) als [Connection](#)

#### Beschreibung

Erstellt zwischen den beiden bereitgestellten Datapoints ([DatapointFrom](#) & [DatapointTo](#)) eine Verbindung.

Es kann keine Verbindung erstellt werden, wenn DatapointFrom kein Datapoint auf der Output-Seite ist, DatapointTo keine Datapoint auf der Input-Seite ist oder bereits eine Verbindung zwischen diesen beiden Datapoints besteht.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1241 Die Verbindung konnte nicht hergestellt werden.

**InsertFunctionCall**

**Methode:** [InsertFunctionCall](#)( [strFunctionName](#) als String, [strLibraryName](#) als String ) als [Component](#)

**Beschreibung**

Fügt eine Funktionsaufruf-Komponente in das aktuelle Mapping ein.

Die angegebenen Bibliotheks- und Funktionsnamen geben die aufzurufende Funktion bzw. benutzerdefinierte Funktion an.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1242 Die Funktionsaufruf-Komponente konnte nicht erstellt werden.

**InsertXMLFile**

**Methode:** [InsertXMLFile](#)( [i\\_strFileName](#) als String, [i\\_strSchemaFileName](#) als String ) als [Component](#)

**Beschreibung**

Fügt eine neue XML-Schemakomponente zum Mapping hinzu.

Die interne Struktur der Komponente wird vom Schema bestimmt, das in der angegebenen XML-Datei ([i\\_strFileName](#)) referenziert wird oder, wenn die XML-Datei keine Schemadatei referenziert, von einer separat definierten Schemadatei ([i\\_strSchemaFileName](#)).

Wenn die XML-Datei eine Schemadatei referenziert, wird der Parameter [i\\_strSchemaFileName](#) ignoriert.

Das Root-Element der XML-Datei wird in der Komponente verwendet.

Die angegebene XML-Datei wird als Input-Beispieldatei zur Überprüfung des Mappings verwendet.

**Fehler**

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1244 Die Komponente konnte nicht erstellt werden.

## InsertXMLSchema

**Methode:** `InsertXMLSchema( i_strSchemaFileName als String, i_strXMLRootName als String )`  
als [Component](#)

### Beschreibung

Fügt eine neue XML-Schemakomponente zum Mapping hinzu.

Die interne Struktur der Komponente wird vom Schema bestimmt, das in der Schemadatei definiert ist.

Der zweite Parameter definiert das Root-Element dieses Schemas, wenn mehr als eines zur Auswahl steht.

Wenn das übergebene Root-Element ein leerer String ist und mehr Auswahlmöglichkeiten zur Verfügung stehen, wird das Dialogfeld "Root-Element auswählen" angezeigt, wenn MapForce sichtbar ist. Wenn MapForce im Hintergrund läuft, wird kein Dialogfeld angezeigt und es wird eine Fehlermeldung zurückgegeben.

Dieser Komponente ist keine Input-XML-Beispieldatei zugewiesen.

### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1244 Die Komponente konnte nicht erstellt werden.

## InsertXMLSchemaInputParameter

**Methode:** `InsertXMLSchemaInputParameter( strParamName als String, strSchemaFileName als String, strXMLRootElementName als String )` als [Component](#)

### Beschreibung

Fügt eine XML-Schema-Input-Parameterkomponente in das aktuelle Mapping ein.

Beim aktuellen Mapping muss es sich um eine **benutzerdefinierte** Funktion handeln. Ein Versuch ihn (den Schema-Input-Parameter) in das **Hauptmapping** einzufügen, wird fehlschlagen.

**strParamName** ist der Name der zu erstellenden Input-Parameterkomponente und **strSchemaFileName** und **strXMLRootElementName** geben die jeweilige Schemadatei und das Root-Element der Schemadatei an, die verwendet werden müssen.

Wenn das übergebene Root-Element ein leerer String ist und mehr Auswahlmöglichkeiten zur Verfügung stehen, wird das Dialogfeld "Root-Element auswählen" angezeigt, wenn MapForce sichtbar ist. Wenn MapForce im Hintergrund läuft, wird kein Dialogfeld angezeigt und es wird eine Fehlermeldung zurückgegeben.

### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1243 Die Parameterkomponente konnte nicht erstellt werden.
- 1245 Diese Operation wird für das Hauptmapping nicht unterstützt.



### InsertXMLSchemaOutputParameter

**Methode:** [InsertXMLSchemaOutputParameter](#)( [strParamName](#) als String, [strSchemaFileName](#) als String, [strXMLRootElementName](#) als String ) als [Component](#)

#### Beschreibung

Fügt einen XML-Schema-Output-Parameter in das aktuelle Mapping ein.

Beim aktuellen Mapping muss es sich um eine **benutzerdefinierte** Funktion handeln. Ein Versuch ihn (den Schema-Output-Parameter) in das **Hauptmapping** einzufügen, wird fehlschlagen.

**strParamName** ist der Name der zu erstellenden Output-Parameterkomponente und **strSchemaFileName** und **strXMLRootElementName** geben die jeweilige Schemadatei und das Root-Element der Schemadatei an, die verwendet werden müssen.

Wenn das übergebene Root-Element ein leerer String ist und mehr Auswahlmöglichkeiten zur Verfügung stehen, wird das Dialogfeld "Root-Element auswählen" angezeigt, wenn MapForce sichtbar ist. Wenn MapForce im Hintergrund läuft, wird kein Dialogfeld angezeigt und es wird eine Fehlermeldung zurückgegeben.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1240 Eine Änderung des Dokuments ist nicht gestattet. Es ist schreibgeschützt.
- 1243 Die Parameterkomponente konnte nicht erstellt werden.
- 1245 Diese Operation wird für das Hauptmapping nicht unterstützt.

### IsMainMapping

**Eigenschaft:** [IsMainMapping](#) als Boolean (schreibgeschützt)

#### Beschreibung

Gibt an, ob das aktuelle Mapping das Hauptmapping des Dokuments ist, in dem sich das Mapping befindet.

True bedeutet, es ist das Hauptmapping.

False bedeutet, es ist eine benutzerdefinierte Funktion (UDF).

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### Name

**Eigenschaft:** [Name](#) als String (schreibgeschützt)

#### Beschreibung

Der Name des Mappings / der benutzerdefinierten Funktion (UDF).

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** [Parent](#) als [Document](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.15 Mappings

Stellt eine Sammlung von Mapping-Objekten dar.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Iterieren durch die Sammlung:

[Count](#)

[Item](#)

#### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

#### Beschreibung

Ruft das oberste Objekt der Applikation ab.

#### Fehler

- 1200 Das application-Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Count

**Eigenschaft:** [Count](#) als [Integer](#) (schreibgeschützt)

#### Beschreibung

Ruft die Anzahl der Mappings in der Sammlung ab.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Item

**Eigenschaft:** [Item](#) ([nIndex](#) als [Integer](#)) als [Mapping](#) (schreibgeschützt)

#### Beschreibung

Ruft das Mapping am Index [nIndex](#) aus der Sammlung ab. Indizes beginnen mit 1.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

#### Parent

**Eigenschaft:** [Parent](#) als [Document](#) (schreibgeschützt)

#### Beschreibung

Das Parent-Objekt gemäß dem Objektmodell.

#### Fehler

- 1200 Das Objekt ist nicht mehr gültig.
- 1201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

## 32.2.16 Optionen

Über dieses Objekt haben Sie Zugriff auf alle im Dialogfeld **Extras | Optionen** verfügbaren MapForce Optionen.

### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Allgemeine Optionen:

[ShowLogoOnPrint](#)

[ShowLogoOnStartup](#)

[UseGradientBackground](#)

Optionen für die Codegenerierung:

[CompatibilityMode](#)

[DefaultOutputEncoding](#)

[DefaultOutputByteOrder](#)

[DefaultOutputByteOrderMark](#)

[XSLTDefaultOutputDirectory](#)

[CodeDefaultOutputDirectory](#)

[CPPSettings\\_DOMType](#)

[CPPSettings\\_GenerateVC6ProjectFile](#)

[CppSettings\\_GenerateVSProjectFile](#)

[CPPSettings\\_LibraryType](#)

[CPPSettings\\_UseMFC](#)

[CSharpSettings\\_ProjectType](#)

### Application

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

### Beschreibung

Ruft das oberste Objekt der Applikation ab.

### Fehler

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### CodeDefaultOutputDirectory

**Eigenschaft:** [CodeDefaultOutputDirectory](#) as String

### Beschreibung

Definiert das Zielverzeichnis, in das von [Document.GenerateCppCode](#), [Document.GenerateJavaCode](#) und [Document.GenerateCHashCode](#) generierte Dateien platziert werden.

### Fehler

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### Siehe auch

Codegenerierung

**CompatibilityMode**

**Eigenschaft:** [CompatibilityMode](#) als Boolean

**Beschreibung**

Setzen Sie die Eigenschaft auf true, um Code zu generieren, der mit Version 2005R3 kompatibel ist. Setzen Sie die Eigenschaft auf false, um die neu hinzugefügten Codegenerierungsfunktionen in [Document.GenerateCppCode](#), [Document.GenerateCHashCode](#), [Document.GenerateJavaCode](#) und [Document.GenerateXSLT](#) zu verwenden.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**CppSettings\_DOMType**

**Eigenschaft:** [CppSettings\\_DOMType](#) als [ENUMDOMType](#)

**Beschreibung**

Definiert den von [Document.GenerateCppCode](#) verwendeten DOM-Typ.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1402 Der Parameter befindet sich außerhalb des Geltungsbereichs
- 1403 Der Parameterwert steht nicht mehr zur Verfügung

**Siehe auch**

Codegenerierung

**CppSettings\_GenerateVC6ProjectFile**

**Eigenschaft:** [CppSettings\\_GenerateVC6ProjectFile](#) als Boolean

**Beschreibung**

Definiert, ob VisualC++ 6.0 Projektdateien von [Document.GenerateCppCode](#) generiert werden sollen.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**CppSettings\_GenerateVSProjectFile**

**Eigenschaft:** [CSharpSettings\\_GenerateVSProjectFile](#) als [ENUMProjectType](#)

**Beschreibung**

Definiert, welche Version von VisualStudio Projektdateien (2005 / 2008 / 2010) von [Document.GenerateCppCode](#) generiert werden soll.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1402 Der Parameter befindet sich außerhalb des Geltungsbereichs
- 1403 Der Parameterwert steht nicht mehr zur Verfügung

**Siehe auch**

Codegenerierung

**CppSettings\_LibraryType**

**Eigenschaft:** [CppSettings\\_LibraryType](#) als [ENUMLibType](#)

**Beschreibung**

Definiert den von [Document.GenerateCppCode](#) verwendeten Bibliothekstyp.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**CppSettings\_UseMFC**

**Eigenschaft:** [CppSettings\\_UseMFC](#) als Boolean

**Beschreibung**

Definiert, ob von C++-Code, der von [Document.GenerateCppCode](#) generiert wurde, MFC-Unterstützung verwendet werden soll.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**CSharpSettings\_ProjectType**

**Eigenschaft:** [CSharpSettings\\_ProjectType](#) als [ENUMProjectType](#)

**Beschreibung**

Definiert die Art des von [Document.GenerateCHashCode](#) verwendeten C#-Projekts.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1402 Der Parameter befindet sich außerhalb des Geltungsbereichs
- 1403 Der Parameterwert steht nicht mehr zur Verfügung

**Siehe auch**

Codegenerierung

**DefaultOutputByteOrder**

**Eigenschaft:** [DefaultOutputByteOrder](#) als String

**Beschreibung**

Bytefolge für die für Ausgabedateien verwendete Dateikodierung.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**DefaultOutputByteOrderMark**

**Eigenschaft:** [DefaultOutputByteOrderMark](#) als Boolean

**Beschreibung**

Gibt an, ob eine Bytefolgemarkierung (Byte Order Mark = BOM) in die Dateikodierung der Ausgabedateien inkludiert werden soll.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**DefaultOutputEncoding**

**Eigenschaft:** [DefaultOutputEncoding](#) als Boolean

**Beschreibung**

Für Ausgabedateien verwendete Dateikodierung.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**GenerateWrapperClasses**

**Eigenschaft:** [GenerateWrapperClasses](#) als Boolean

**Beschreibung**

Gibt an, ob bei der Codegenerierung auch Wrapper-Klassen generiert werden sollen.

**Fehler**



- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**JavaSettings\_ApacheAxisVersion (nicht mehr verwendet)**

**Eigenschaft:** [JavaSettings\\_ApacheAxisVersion](#) als [ENUMApacheAxisVersion](#)

**Beschreibung**

Definiert, welche Apache Axis-Version beim Generieren von Java-Code für Webservice-Implementierungen mit SOAP 1.1 verwendet werden soll.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**Parent**

**Eigenschaft:** [Parent](#) als [Application](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt laut Objektmodell.

**Fehler**

- 1400 Das Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ShowLogoOnPrint**

**Eigenschaft:** [ShowLogoOnPrint](#) als Boolean

**Beschreibung**

MapForce Logo in der Druckausgabe ein- oder ausblenden.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**ShowLogoOnStartup**

**Eigenschaft:** [ShowLogoOnStartup](#) als Boolean

**Beschreibung**

MapForce Logo beim Start der Applikation ein- oder ausblenden.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**UseGradientBackground**

**Eigenschaft:** [UseGradientBackground](#) als Boolean

**Beschreibung**

Hintergrundfarbmodus für ein Mapping-Fenster definieren oder abrufen.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**WrapperClassesVersion**

**Eigenschaft:** [WrapperClassesVersion](#) als [ENUMWrapperClassesVersion](#)

**Beschreibung**

Definiert, welche Wrapper Class-Version beim Generieren von Code erstellt werden soll, wenn die Generierung von Wrapper-Klassen aktiviert ist.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

**XSLTDefaultOutputDirectory**

**Eigenschaft:** [XSLTDefaultOutputDirectory](#) als String

**Beschreibung**

Definiert das Zielverzeichnis, in das von [Document.GenerateXSLT](#) generierte Dateien platziert werden sollen.

**Fehler**

- 1400 Das application-Objekt ist nicht mehr gültig.
- 1401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Siehe auch**

Codegenerierung

### 32.2.17 Projekt (Enterprise oder Professional Edition)

Ein Projekt-Objekt repräsentiert ein Projekt und seine Projektelementstruktur in MapForce.

#### Events

[Events](#)

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Dateibehandlung:

[FullName](#)

[Name](#)

[Path](#)

[Saved](#)

[Save](#)

[Close](#)

Navigation in der Projektstruktur:

[Count](#)

[Item](#)

[NewEnum](#)

Bearbeitung der Projektstruktur:

[AddActiveFile](#)

[AddFile](#)

[InsertWebService](#) (nur Enterprise Edition)

[CreateFolder](#)

Codegenerierung:

[Output\\_Folder](#)

[Output\\_Language](#)

[Output\\_TextEncoding](#)

[Java\\_BasePackageName](#)

[GenerateCode](#)

[GenerateCodeEx](#)

[GenerateCodeIn](#)

[GenerateCodeInEx](#)

Beispiele zur Verwendung der oben aufgelisteten Eigenschaften und Methoden finden Sie unter [Beispiel: Projektunterstützung](#). Beachten Sie: Um diese Eigenschaften und Methoden verwenden zu können, muss die Enterprise oder Professional Edition von MapForce auf Ihrem Computer installiert sein. Für Operationen mit Webservices benötigen Sie die Enterprise Edition.

#### Events

Dieses Objekt unterstützt die folgenden Events:

[OnProjectClosed](#)

OnProjectClosed

**Event:** [OnProjectClosed](#) (*i\_objProject* als [Project](#))

**Beschreibung**

Dieses Event wird beim Schließen des Projekts ausgelöst. Das an den Event Handler übergebene Projektobjekt sollte nicht aufgerufen werden. Das entsprechende Event zum Öffnen ist [Application.OnProjectOpened](#).

**\_NewEnum**

**Eigenschaft:** [\\_NewEnum](#) () als IUnknown (schreibgeschützt)

**Beschreibung**

Diese Eigenschaft unterstützt die sprachspezifische Standard-Enumeration.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

**Beispiele**

```
// -----
// JScript sample - enumeration of a project's project items.
function AllChildrenOfProjectRoot()
{
 objProject = objMapForce.ActiveProject;
 if (objProject != null)
 {
 for (objProjectIter = new Enumerator(objProject); !
objProjectIter.atEnd(); objProjectIter.moveNext())
 {
 objProjectItem = objProjectIter.item();

 // do something with project item here
 }
 }
}

// -----
// JScript sample - iterate all project items, depth first.
function IterateProjectItemsRec(objProjectItemIter)
{
 while (! objProjectItemIter.atEnd())
 {
 objProjectItem = objProjectItemIter.item();
 // do something with project item here

 IterateProjectItemsRec(new Enumerator(objProjectItem));

 objProjectItemIter.moveNext();
 }
}
function IterateAllProjectItems()
{
 objProject = objMapForce.ActiveProject;
 if (objProject != null)
 {
 IterateProjectItemsRec(new Enumerator(objProject));
 }
}
```

**AddActiveFile**

**Methode:** [AddActiveFile](#) () als [ProjectItem](#)

**Beschreibung**

Fügt das gerade offene Dokument zum Mapping-Ordner der Root des Projekts hinzu.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1503 Es ist kein aktives Dokument verfügbar.
- 1504 Das aktive Dokument muss einen Pfadnamen erhalten, bevor es zum Projekt hinzugefügt werden kann.
- 1705 Das Mapping konnte dem Projekt nicht zugewiesen werden.  
Möglicherweise ist es bereits im Zielordner vorhanden.

**AddFile**

**Methode:** `AddFile` (*`i_strFileName`* als String) als [ProjectItem](#)

**Beschreibung**

Fügt das angegebene Dokument zum Mapping-Ordner oder zur Root des Projekts hinzu.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Der Dateiname ist leer.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1705 Das Mapping konnte dem Projekt nicht zugewiesen werden.  
Entweder existiert die Datei nicht oder sie ist kein MapForce Mapping.  
Möglicherweise ist die Datei bereits im Zielordner vorhanden.

**Application**

**Eigenschaft:** `Application` als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das Applikationsobjekt auf oberster Ebene auf.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Close**

**Methode:** `Close` ()

**Beschreibung**

Schließt das Projekt ohne es zu speichern.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.

**Count**

**Eigenschaft:** `Count` als Integer (schreibgeschützt)

**Beschreibung**

Ruft die Anzahl der Children des Root-Elements des Projekts ab.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

**Examples**

Siehe [Item](#) oder [NewEnum](#).

**CreateFolder**

**Methode:** [CreateFolder](#) (*i\_strFolderName* als String) als [ProjectItem](#)

**Beschreibung**

Erstellt einen neuen Ordner als Child des Root-Elements des Projekts.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

1501 Ungültiger Ordnername oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

**FullName**

**Eigenschaft:** [FullName](#) als String (schreibgeschützt)

**Beschreibung**

Pfad und Name der Projektdatei.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**GenerateCode**

**Methode:** [GenerateCode](#) ()

**Beschreibung**

Generiert Code für alle Projektelemente des Projekts. Die Codesprache und der Ausgabepfad werden von den Eigenschaften des Projekts und der Projektelemente bestimmt.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

1706 Fehler bei der Codegenerierung

**GenerateCodeEx**

**Methode:** [GenerateCode](#) () als [ErrorMarkers](#)

**Beschreibung**

Generiert Code für alle Projektelemente des Projekts. Die Codesprache und der Ausgabepfad werden von den Eigenschaften des Projekts und der Projektelemente bestimmt. Ein Objekt wird zurückgegeben, das verwendet werden kann, um durch alle bei der Codegenerierung generierten Meldungen zu iterieren. Diese Meldungen sind dieselben, wie die, die im Fenster *Meldungen* von MapForce angezeigt werden.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1706 Fehler bei der Codegenerierung

### GenerateCodeIn

**Methode:** `GenerateCodeIn` (*`i_nLanguage`* als [ENUMProgrammingLanguage](#))

#### Beschreibung

Generiert Code für alle Projektelemente in der angegebenen Sprache. Der Ausgabepfad wird von den Eigenschaften des Projekts und der Projektelemente bestimmt.

#### Fehler

- 1500 Das Objekt ist nicht mehr gültig.
- 1706 Fehler bei der Codegenerierung

### GenerateCodeInEx

**Methode:** `GenerateCodeIn` (*`i_nLanguage`* als [ENUMProgrammingLanguage](#)) as [ErrorMarkers](#)

#### Beschreibung

Generiert Code für alle Projektelemente des Projekts in der angegebenen Sprache. Der Ausgabepfad wird von den Eigenschaften des Projekts und der Projektelemente bestimmt. Ein Objekt wird zurückgegeben, das verwendet werden kann, um durch alle bei der Codegenerierung generierten Meldungen zu iterieren. Diese Meldungen sind dieselben, wie die, die im Fenster *Meldungen* von MapForce angezeigt werden.

Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the *Messages* window of MapForce.

#### Fehler

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1706 Fehler bei der Codegenerierung

### InsertWebService

**Methode:** `InsertWebService` (*`i_strWSDLFile`* als String, *`i_strService`* als String, *`i_strPort`* als String, *`i_bGenerateMappings`* als Boolean) als [ProjectItem](#)

#### Beschreibung

Fügt ein neues Webservice Projekt in den Webservice-Ordner des Projekts ein. Wenn *i\_bGenerateMappings* true ist, werden die Mapping-Anfangsdokumente für alle Ports automatisch generiert.

#### Fehler

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 WSDL-Datei kann nicht gefunden werden oder ist ungültig.  
Service- oder Port-Namen sind ungültig.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1503 Die Operation wird von der aktuellen Version nicht unterstützt.

**Item**

**Eigenschaft:** `Item(i_nItemIndex als Integer )` als [ProjectItem](#) (schreibgeschützt)

**Beschreibung**

Gibt das Child an der Position `i_nItemIndex` der Projekt-Root zurück. Der Index ist nullbasiert. Der größte gültige Index ist [Count](#) - 1. Eine alternative Methode zum Aufrufen aller Children finden Sie unter [NewEnum](#).

**Fehler**

1500 Das Objekt ist nicht mehr gültig.

**Examples**

```
// -----
// JScript code snippet - enumerate children using Count and Item.
for(nItemIndex = 0; nItemIndex < objProject.Count; nItemIndex++)
{
 objProjectItem = objProject.Item(nItemIndex);
 // do something with project item here
}
```

**Java\_BasePackageName**

**Eigenschaft:** `Java_BasePackageName` als String

**Beschreibung**

Definiert den Basispaketnamen der Java-Pakete, die generiert werden, bzw. ruft diesen ab. Diese Eigenschaft wird nur beim Generieren von Java-Code verwendet.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.  
 1501 Invalid package name specified.  
 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Name**

**Eigenschaft:** `Name` als String (schreibgeschützt)

**Beschreibung**

Name der Projektdatei ohne Dateipfad.

**Fehler**

1500 Das Objekt ist nicht mehr gültig.  
 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Output\_Folder**

**Eigenschaft:** `Output_Folder` als String

**Beschreibung**

Definiert den Standardausgabeordner, der mit [GenerateCode](#) und [GenerateCodeIn](#) verwendet wird, bzw. ruft diesen ab. Projektelemente können diesen Wert in ihrer Eigenschaft [CodeGenSettings\\_OutputFolder](#) überschreiben, wenn [CodeGenSettings\\_UseDefault](#) auf false gesetzt wurde.



**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Es wurde ein ungültiger Ordnername definiert.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Output\_Language**

**Eigenschaft:** `Output_Language` als [ENUMProgrammingLanguage](#)

**Beschreibung**

Definiert die Standardsprache für die Codegenerierung bei Verwendung von [GenerateCode](#) bzw. ruft diese ab. Projektelemente können diesen Wert in ihrer Eigenschaft [CodeGenSettings\\_OutputLanguage](#) überschreiben, wenn [CodeGenSettings\\_UseDefault](#) auf false gesetzt wurde.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Es wurde eine ungültige Sprache definiert.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Output\_TextEncoding**

**Eigenschaft:** `Output_TextEncoding` als `String`

**Beschreibung**

Definiert die beim Generieren von XML-basiertem Code verwendete Textcodierung bzw. ruft diese ab.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Es wurde eine ungültige Textcodierung definiert.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Parent**

**Eigenschaft:** `Parent` als [Application](#) (schreibgeschützt)

**Beschreibung**

Das Parent-Objekt gemäß dem Objektmodell.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Path**

**Eigenschaft:** `Path` als `String` (schreibgeschützt)

**Beschreibung**

Pfad der Projektdatei ohne Namen.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Save**

**Methode:** [Save](#) ()

**Beschreibung**

Speichert das Projekt in der durch [FullName](#) definierten Datei.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1502 Kann nicht in Datei gespeichert werden.

**Saved**

**Eigenschaft:** [Saved](#) als Boolean (schreibgeschützt)

**Beschreibung**

True, wenn das Projekt seit der letzten Speicherung nicht geändert wurde, andernfalls false.

**Fehler**

- 1500 Das Objekt ist nicht mehr gültig.
- 1501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### 32.2.18 ProjectItem (Enterprise oder Professional Edition)

Ein ProjectItem-Objekt repräsentiert einen Eintrag in der Projektstruktur.

#### Eigenschaften und Methoden

Eigenschaften zum Navigieren im Objektmodell:

[Application](#)

[Parent](#)

Navigation in der Projektstruktur:

[Count](#)

[Item](#)

[\\_NewEnum](#)

Projektelementeigenschaften:

[Kind](#)

[Name](#)

[WSDLFile](#) (steht nur bei Webservice-Projektelementen zur Verfügung)

[QualifiedNames](#) (steht nur bei Webservice-Projektelementen zur Verfügung)

Bearbeitung der Projektstruktur:

[AddActiveFile](#) (steht nur für Ordner Elemente zur Verfügung)

[AddFile](#) (steht nur für Ordner Elemente zur Verfügung)

[CreateFolder](#) (steht nur für Ordner Elemente zur Verfügung)

[CreateMappingForProject](#) (steht nur bei Webservice-Operationen zur Verfügung)

[Remove](#)

Dokumentzugriff:

[Open](#) (steht nur für Mapping-Elemente und Webservice-Operationen zur Verfügung)

Codegenerierung:

[CodeGenSettings\\_UseDefault](#)

[CodeGenSettings\\_OutputFolder](#)

[CodeGenSettings\\_Language](#)

[GenerateCode](#)

[GenerateCodeEx](#)

[GenerateCodeIn](#)

[GenerateCodeInEx](#)

Beispiele zur Verwendung der oben aufgelisteten Eigenschaften und Methoden finden Sie unter [Beispiel: Projektunterstützung](#). Beachten Sie: Um diese Eigenschaften und Methoden verwenden zu können, müssen Sie die Enterprise oder Professional Edition von MapForce installiert haben. Für Operationen mit Webservices benötigen Sie die Enterprise Edition.

#### [\\_NewEnum](#)

**Eigenschaft:** [\\_NewEnum](#) () als IUnknown (schreibgeschützt)

#### Beschreibung

Diese Eigenschaft unterstützt sprachspezifische Standardenumeration.

#### Fehler

1700 Das Objekt ist nicht mehr gültig.

#### Beispiele

Siehe [Project.Item](#) oder [Project.\\_NewEnum](#).

**AddActiveFile**

**Methode:** [AddActiveFile](#) () als [ProjectItem](#)

**Beschreibung**

Fügt das gerade aktive Dokument zu diesem Projektelement hinzu, wenn es ein gültiges Child ist. Andernfalls wird das Dokument zum Mapping-Ordner der Projekt-Root hinzugefügt.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Der Dateiname ist leer.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1703 Es ist kein aktives Dokument verfügbar.
- 1704 Das aktive Dokument muss einen Pfadnamen erhalten, bevor es zum Projekt hinzugefügt werden kann.
- 1705 Das Mapping konnte dem Projekt nicht zugewiesen werden.  
Entweder die Datei existiert nicht oder ist kein MapForce Mapping.  
Möglicherweise wurde die Datei dem Zielordner bereits zugewiesen.

**AddFile**

**Methode:** [AddFile](#) ([i\\_strFileName](#) als String) als [ProjectItem](#)

**Beschreibung**

Fügt das angegebene Dokument zu diesem Projektelement hinzu, wenn es ein gültiges Child-Dokument ist. Andernfalls wird das Dokument zum Mapping-Ordner der Projekt-Root hinzugefügt.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Der Dateiname ist leer.  
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1705 Das Mapping konnte dem Projekt nicht zugewiesen werden.  
Entweder die Datei existiert nicht oder ist kein MapForce Mapping.  
Möglicherweise wurde die Datei dem Zielordner bereits zugewiesen.

**Application**

**Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)

**Beschreibung**

Ruft das oberste Applikationsobjekt ab.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**CodeGenSettings\_Language**

**Eigenschaft:** [CodeGenSettings\\_Language](#) als [ENUMProgrammingLanguage](#)

**Beschreibung**

Definiert die mit [GenerateCode](#) oder [Project.GenerateCode](#) zu verwendende Sprache oder ruft diese ab. Diese Eigenschaft wird nur benötigt, wenn [CodeGenSettings UseDefault](#) auf false gesetzt ist.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Es wurde für den Rückgabeparameter eine ungültige Sprache oder eine ungültige Adresse angegeben.

**CodeGenSettings\_OutputFolder**

**Eigenschaft:** [CodeGenSettings\\_OutputFolder](#) als String

**Beschreibung**

Definiert das mit [GenerateCode](#), [GenerateCodeIn](#), [Project.GenerateCode](#) oder [Project.GenerateCodeIn](#) zu verwendende Ausgabeverzeichnis oder ruft dieses ab. Diese Eigenschaft wird nur benötigt, wenn [CodeGenSettings\\_UseDefault](#) auf false gesetzt ist.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Es wurde für den Rückgabeparameter ein ungültiger Ordner oder eine ungültige Adresse angegeben.

**CodeGenSettings\_UseDefault**

**Eigenschaft:** [CodeGenSettings\\_UseDefault](#) als Boolean

**Beschreibung**

Definiert bzw. ruft ab, ob das Ausgabeverzeichnis und die Codesprache wie von (a) den Parent-Ordern oder (b) der Projekt-Root definiert, verwendet werden sollen. Diese Eigenschaft wird bei Aufrufen von [GenerateCode](#), [GenerateCodeIn](#), [Project.GenerateCode](#) und [Project.GenerateCodeIn](#) verwendet. Wenn diese Eigenschaft auf false gesetzt ist, werden die Werte von [CodeGenSettings\\_OutputFolder](#) und [CodeGenSettings\\_Language](#) verwendet, um Code für dieses Projektelement zu generieren.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Count**

**Eigenschaft:** [Count](#) als Integer (schreibgeschützt)

**Beschreibung**

Ruft die Anzahl der Children dieses Projektelements ab. Siehe auch [Item](#).

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.

**Examples**

Siehe [Project.Item](#) oder [Project.NewEnum](#).

**CreateFolder**

**Methode:** [CreateFolder](#) (*i\_strFolderName* als String) als [ProjectItem](#)

**Beschreibung**

Erstellt einen neuen Ordner als Child dieses Projektelements.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde ein ungültiger Ordnername oder eine ungültige Adresse angegeben.
- 1702 Das Projektelement unterstützt keine Child-Elemente.

**CreateMappingForProject**

**Methode:** `CreateMappingForProject` (*i\_strFileName* als String) als [ProjectItem](#)

**Beschreibung**

Erstellt ein Mapping-Anfangsdokument für eine Webservice-Operation und speichert es unter *i\_strFileName*. Bei Verwendung von [Project.InsertWebService](#) können Sie das Flag *i\_bGenerateMappings* verwenden, damit MapForce automatisch Anfangsmappings für alle Ports erstellt.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1707 Neues Mapping kann nicht erstellt werden.  
Das Projektelement unterstützt entweder die automatische Erstellung von Anfangsmappings nicht oder es ist bereits ein Mapping vorhanden.
- 1708 Die Operation wird in der aktuellen Version nicht unterstützt.

**GenerateCode**

**Methode:** `GenerateCode` ()

**Beschreibung**

Generiert Code für dieses Projektelement und seine Children. Die Codesprache und das Ausgabeverzeichnis werden durch [CodeGenSettings UseDefault](#), [CodeGenSettings Language](#) und [CodeGenSettings OutputFolder](#) definiert. Child-Elemente dieses Projektelements können ihre eigenen Eigenschaftseinstellungen für die Codegenerierung haben.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1706 Fehler bei der Codegenerierung.

**GenerateCodeEx**

**Methode:** `GenerateCode` () als [ErrorMarkers](#)

**Beschreibung**

Generiert Code für dieses Projektelement und seine Child-Elemente. Die Codesprache und das Ausgabeverzeichnis werden durch [CodeGenSettings UseDefault](#), [CodeGenSettings Language](#) und [CodeGenSettings OutputFolder](#) definiert. Child-Elemente dieses Projektelements können ihre eigenen Eigenschaftseinstellungen für die Codegenerierung haben.

Bei der Codegenerierung wird ein Objekt zurückgegeben, mit dem Sie durch alle ausgegebenen Meldungen iterieren können. Diese Meldungen sind dieselben wie die im Fenster *Meldungen* von MapForce angezeigten.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1706 Fehler bei der Codegenerierung.

**GenerateCodeIn**

**Methode:** `GenerateCodeIn` (*`i_nLanguage`* als [ENUMProgrammingLanguage](#))

**Beschreibung**

Generiert in der angegebenen Sprache Code für dieses Projektelement und seine Child-Elemente. Das Ausgabeverzeichnis wird durch [CodeGenSettings UseDefault](#) und [CodeGenSettings OutputFolder](#) definiert. Child-Elemente dieses Projektelements können ihre eigenen Eigenschaftseinstellungen für die Codegenerierung haben.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Es wurde eine ungültige Sprache definiert.
- 1706 Fehler bei der Codegenerierung.

**GenerateCodeInEx**

**Methode:** `GenerateCodeIn` (*`i_nLanguage`* als [ENUMProgrammingLanguage](#)) als [ErrorMarkers](#)

**Beschreibung**

Generiert in der angegebenen Sprache Code für dieses Projektelement und seine Child-Elemente. Das Ausgabeverzeichnis wird durch [CodeGenSettings UseDefault](#) und [CodeGenSettings OutputFolder](#) definiert. Child-Elemente dieses Projektelements können ihre eigenen Eigenschaftseinstellungen für die Codegenerierung haben.

Bei der Codegenerierung wird ein Objekt zurückgegeben, mit dem Sie durch alle ausgegebenen Meldungen iterieren können. Diese Meldungen sind dieselben wie die im Fenster *Meldungen* von MapForce angezeigt.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Es wurde eine ungültige Sprache definiert oder für den Rückgabeparameter wurde eine ungültige Adresse definiert.
- 1706 Fehler bei der Codegenerierung.

**Item**

**Eigenschaft:** `Item` (*`i_nItemIndex`* als Integer) als [ProjectItem](#) (schreibgeschützt)

**Beschreibung**

Gibt das Child an der Position `i_nItemIndex` dieses Projektelements zurück. Der Index ist nullbasiert. Der größte Wert ist [Count](#) - 1.  
Eine andere Methode zum Aufrufen aller Child-Elemente finden Sie unter [\\_NewEnum](#).

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.

**Examples**

Siehe [Project.Item](#) oder [Project.NewEnum](#).

**Kind**

**Eigenschaft:** [Kind](#) als [ENUMProjectItemType](#) (schreibgeschützt)

**Beschreibung**

Ruft die Art des Projektelements ab. Die Verfügbarkeit einiger Eigenschaften und die Anwendbarkeit bestimmter Methoden ist auf bestimmte Arten von Projektelementen beschränkt. Die Beschreibung aller Methoden und Eigenschaften enthält Informationen über diese Einschränkungen.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

**Name**

**Eigenschaft:** [Name](#) als String

**Beschreibung**

Ruft den Namen eines Projektelements ab oder definiert diesen. Der Name der meisten Elemente ist schreibgeschützt. Ausnahmen sind vom Benutzer erstellte Ordner. Die Namen solcher Ordner können nach der Erstellung geändert werden.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1702 Der Name des Projektelements kann nicht geändert werden.

**Open**

**Methode:** [Open](#) () als [Document](#)

**Beschreibung**

Öffnet das Projektelement als Dokument oder macht das entsprechende Dokument zum aktiven, wenn es bereits geöffnet ist. Das Projektelement muss ein MapForce Mapping sein oder - nur bei der Enterprise Edition - eine Webservice-Operation.

**Fehler**

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1702 Das Projektelement bezieht sich auf keine MapForce Mapping-Datei.
- 1708 Die Operation wird in der aktuellen Version nicht unterstützt.

**Parent**

**Eigenschaft:** [Parent](#) als [Project](#) (schreibgeschützt)

**Beschreibung**

Ruft das Projekt ab, von dem dieses Element ein Child-Element ist. Hat dieselbe Wirkung wie `Application.ActiveProject`.

**Fehler**



- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

### QualifiedName

**Eigenschaft:** `QualifiedName` als `String` (schreibgeschützt)

#### Beschreibung

Ruft den qualifizierten Namen eines Webservice-Elements ab.

#### Fehler

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1702 Das Projektelement ist nicht Teil eines Webservice.

### Remove

**Methode:** `Remove()`

#### Beschreibung

Entfernt dieses Projektelement und alle seine Child-Elemente aus der Projektstruktur.

#### Fehler

- 1700 Das Objekt ist nicht mehr gültig.

### WSDLFile

**Eigenschaft:** `WSDLFile` als `String` (schreibgeschützt)

#### Beschreibung

Ruft den Dateinamen der WSDL-Datei ab, die den Webservice definiert, der das aktuelle Projektelement enthält.

#### Fehler

- 1700 Das Objekt ist nicht mehr gültig.
- 1701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1702 Das Projektelement ist nicht Teil eines Webservice.

## 32.3 Enumerationen

Dies ist eine Liste aller von der MapForce API verwendeten Enumerationen. Wenn Ihre Scripting-Umgebung Enumerationen nicht unterstützt, verwenden Sie statt dessen die numerischen Werte.

### 32.3.1 ENUMApacheAxisVersion (nicht mehr verwendet)

**Beschreibung**

Enumerationswerte zur Auswahl der Apache Axis-Version.

**Mögliche Werte:**

|                          |     |
|--------------------------|-----|
| eApacheAxisVersion_Axis  | = 1 |
| eApacheAxisVersion_Axis2 | = 2 |

**Siehe auch**

Codegenerierung

### 32.3.2 ENUMApplicationStatus

**Beschreibung**

Enumerationswerten zur Angabe des Status der Applikation.

**Mögliche Werte:**

|                                          |     |
|------------------------------------------|-----|
| eApplicationRunning                      | = 0 |
| eApplicationAfterLicenseCheck            | = 1 |
| eApplicationBeforeLicenseCheck           | = 2 |
| eApplicationConcurrentLicenseCheckFailed | = 3 |

### 32.3.3 ENUMAppOutputLine\_Severity

**Beschreibung**

Enumerationswerte zur Angabe des Schweregrads einer AppOutputLine.

**Mögliche Werte:**

|                                          |      |
|------------------------------------------|------|
| eSeverity_Undefined                      | = -1 |
| eSeverity_Info                           | = 0  |
| eSeverity_Warning                        | = 1  |
| eSeverity_Error                          | = 2  |
| eSeverity_CriticalError                  | = 3  |
| eSeverity_Success                        | = 4  |
| eSeverity_Summary                        | = 5  |
| eSeverity_Progress                       | = 6  |
| eSeverity_DataEdit                       | = 7  |
| eSeverity_ParserInfo                     | = 8  |
| eSeverity_PossibleInconsistencyWarning   | = 9  |
| eSeverity_Message                        | = 10 |
| eSeverity_Document                       | = 11 |
| eSeverity_Rest                           | = 12 |
| eSeverity_NoSelect                       | = 13 |
| eSeverity_Select                         | = 14 |
| eSeverity_Autoinsertion                  | = 15 |
| eSeverity_GlobalResources_DefaultWarning | = 16 |

### 32.3.4 ENUMAppOutputLine\_TextDecoration

**Beschreibung**

Enumerationswerte für die verschiedenen Arten von Textdekoration einer AppOutputLine.

**Mögliche Werte:**

|                                         |      |
|-----------------------------------------|------|
| eTextDecorationDefault                  | = 0  |
| eTextDecorationBold                     | = 1  |
| eTextDecorationDebugValues              | = 2  |
| eTextDecorationDB_ObjectName            | = 3  |
| eTextDecorationDB_ObjectLink            | = 4  |
| eTextDecorationDB_ObjectKind            | = 5  |
| eTextDecorationDB_TimeoutValue          | = 6  |
| eTextDecorationFind_MatchingString      | = 7  |
| eTextDecorationValidation_Speclink      | = 8  |
| eTextDecorationValidation_ErrorPosition | = 9  |
| eTextDecorationValidation_UnkownParam   | = 10 |

### 32.3.5 ENUMCodeGenErrorLevel

**Beschreibung**

Enumerationswerte zum Identifizieren der Schwere von Codegenerierungsmeldungen.

**Mögliche Werte:**

```
eCodeGenErrorLevel_Information = 0
eCodeGenErrorLevel_Warning = 1
eCodeGenErrorLevel_Error = 2
eCodeGenErrorLevel_Undefined = 3
```

### 32.3.6 ENUMComponentDatapointSide

Beschreibung

Enumerationswerte zur Angabe der Seite eines Datapoint in seiner Komponente.

**Mögliche Werte:**

|                      |     |
|----------------------|-----|
| eDatapointSideInput  | = 0 |
| eDatapointSideOutput | = 1 |

Siehe auch

[GetRootDatapoint](#)



### 32.3.7 ENUMComponentSubType

**Beschreibung**

Enumerationswerte zur Angabe der Komponentenuntertypen.

**Mögliche Werte:**

|                               |     |
|-------------------------------|-----|
| eComponentSubType_None        | = 0 |
| eComponentSubType_Text_EDI    | = 1 |
| eComponentSubType_Text_Flex   | = 2 |
| eComponentSubType_Text_CSVFLF | = 3 |

### 32.3.8 ENUMComponentType

**Beschreibung**

Enumerationswerte zur Angabe der Komponententypen.

**Mögliche Werte:**

|                        |     |
|------------------------|-----|
| eComponentType_Unknown | = 0 |
| eComponentType_XML     | = 1 |
| eComponentType_DB      | = 2 |
| eComponentType_Text    | = 3 |
| eComponentType_Excel   | = 4 |
| eComponentType_WSDL    | = 5 |
| eComponentType_XBRL    | = 6 |

### 32.3.9 ENUMComponentUsageKind

**Beschreibung**

Enumerationswerte zur Angabe der Verwendungsart der Komponente.

**Mögliche Werte:**

|                              |     |
|------------------------------|-----|
| eComponentUsageKind_Unknown  | = 0 |
| eComponentUsageKind_Instance | = 1 |
| eComponentUsageKind_Input    | = 2 |
| eComponentUsageKind_Output   | = 3 |
| eComponentUsageKind_Variable | = 4 |

### 32.3.10 ENUMConnectionType

**Beschreibung**

Enumerationswerte zur Angabe des Typs einer Verbindung.

**Possible values:**

|                             |     |
|-----------------------------|-----|
| eConnectionTypeTargetDriven | = 0 |
| eConnectionTypeSourceDriven | = 1 |
| eConnectionTypeCopyAll      | = 2 |

Siehe auch

[ConnectionType](#)

### 32.3.11 ENUMDOMType

**Beschreibung**

Enumerationswerte zur Definition des vom generierten C++ Mapping-Code verwendeten DOM-Typs.

**Mögliche Werte:**

|                  |     |
|------------------|-----|
| eDOMType_xerces  | = 1 |
| eDOMType_xerces3 | = 2 |
| eDOMType_msxml6  | = 3 |

Nicht mehr  
verwendete Werte

|                 |     |
|-----------------|-----|
| eDOMType_msxml4 | = 0 |
|-----------------|-----|

"Nicht mehr verwendet" bedeutet in diesem Zusammenhang, dass dieser Wert nicht unterstützt wird und nicht verwendet werden sollte.

eDOMType\_xerces zeigt an, dass Xerces 2.x verwendet wird; eDOMType\_xerces3 gibt an, dass Xerces 3.x verwendet wird.

**Siehe auch**

Codegenerierung

### 32.3.12 ENUMLibType

**Beschreibung**

Enumerationenwerte zur Definition des vom generierten C++ Mapping-Code verwendeten Bibliothekstyps.

**Mögliche Werte:**

|                 |     |
|-----------------|-----|
| eLibType_static | = 0 |
| eLibType_dll    | = 1 |

**Siehe auch**

Codegenerierung

### 32.3.13 ENUMProgrammingLanguage

**Beschreibung**

Enumerationswerte zur Auswahl der Programmiersprache.

**Mögliche Werte:**

|                    |      |
|--------------------|------|
| eUndefinedLanguage | = -1 |
| eJava              | = 0  |
| eCpp               | = 1  |
| eCSharp            | = 2  |
| eXSLT              | = 3  |
| eXSLT2             | = 4  |
| eXQuery            | = 5  |

**Siehe auch**

Codegenerierung

### 32.3.14 ENUMProjectItemType

**WBeschreibung**

Enumeration zur Angabe der verschiedenen Arten von Projektelementen, die Child-Elemente von [Project](#) oder ordnerähnlichen [ProjectItems](#) sind.

**Mögliche Werte:**

|                                      |      |
|--------------------------------------|------|
| eProjectItemType_Invalid             | = -1 |
| eProjectItemType_MappingFolder       | = 0  |
| eProjectItemType_Mapping             | = 1  |
| eProjectItemType_WebServiceFolder    | = 2  |
| eProjectItemType_WebServiceRoot      | = 3  |
| eProjectItemType_WebServiceService   | = 4  |
| eProjectItemType_WebServicePort      | = 5  |
| eProjectItemType_WebServiceOperation | = 6  |
| eProjectItemType_ExternalFolder      | = 7  |
| eProjectItemType_LibraryFolder       | = 8  |
| eProjectItemType_ResourceFolder      | = 9  |
| eProjectItemType_VirtualFolder       | = 10 |

**Siehe auch**

[ProjectItem.Kind](#)



### 32.3.15 ENUMProjectType

**Beschreibung**

Enumerationswerte zur Auswahl eines Projekttyps für den generierten C# Mapping-Code.

**Mögliche Werte:**

|                                |                          |     |                   |
|--------------------------------|--------------------------|-----|-------------------|
|                                | eVisualStudio2005Project | = 4 | Auch für C++ Code |
|                                | eVisualStudio2008Project | = 5 | Auch für C++ Code |
|                                | eVisualStudio2010Project | = 6 | Auch für C++ Code |
| Nicht mehr<br>verwendete Werte | eVisualStudioProject     | = 0 |                   |
|                                | eVisualStudio2003Project | = 1 |                   |
|                                | eBorlandProject          | = 2 |                   |

"Nicht mehr verwendet" bedeutet in diesem Zusammenhang, dass dieser Wert nicht unterstützt wird und nicht verwendet werden sollte.

**Siehe auch**

Codegenerierung

### 32.3.16 ENUMSearchDatapointFlags

**Beschreibung**

Enumerationswerte, die als Bit-Flags verwendet werden; dienen als Kombination von Flags beim Suchen nach einem Datapoint.

**Mögliche Werte:**

|                           |     |
|---------------------------|-----|
| eSearchDatapointElement   | = 1 |
| eSearchDatapointAttribute | = 2 |

Siehe auch

[GetChild](#)

### 32.3.17 ENUMViewMode

**Beschreibung**

Enumerationswerte zur Auswahl einer MapForce-Ansicht.

**Mögliche Werte:**

|               |     |
|---------------|-----|
| eMapForceView | = 0 |
| eXSLView      | = 1 |
| eOutputView   | = 2 |

### 32.3.18 ENUMWrapperClassesVersion

**Beschreibung**

Enumerationenwerte zur Auswahl der Version der generierten Wrapper-Klassen.

**Mögliche Werte:**

|                               |     |
|-------------------------------|-----|
| eWrapperClassesVersion_2005r3 | = 0 |
| eWrapperClassesVersion_2006   | = 1 |
| eWrapperClassesVersion_2007r3 | = 2 |

**Siehe auch**

Codegenerierung

# Kapitel 33

---

## ActiveX Integration

## 33 ActiveX Integration

MapForceControl ist ein Control, das ein Mittel zur Integration der MapForce 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 MapForce zu integrieren, müssen Sie das MapForce Integrationspaket installieren. Stellen Sie sicher, dass Sie MapForce zuerst installieren und dann erst das MapForce Integrationspaket.

Für eine erfolgreiche Integration müssen beim Design die folgenden Hauptfaktoren berücksichtigt werden:

- Welche Technologie oder Programmiersprache kann die Host-Applikation zur Integration des MapForceControl verwenden?
- Soll die integrierte Benutzeroberfläche genau wie MapForce 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 MapForce 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 [MapForce Automation-Schnittstelle](#) auch über das MapForceControl aufgerufen werden.

Informationen zum Integrieren von MapForce in Microsoft Visual Studio finden Sie im Abschnitt MapForce in Visual Studio.

### **MapForce Integration und Deployment auf Client-Rechnern:**

Wenn Sie eine .NET-Applikation erstellen und diese auf anderen Clients zur Verfügung stellen möchten, müssen Sie auf den Client Rechnern die folgenden Komponenten installieren:

- die MapForce Applikation
- das Integrationspaket für MapForce
- den benutzerdefinierten Integrationscode.

## 33.1 Integration auf Applikationsebene

Die Integration auf Applikationsebene ist einfach. Sie können dadurch die gesamte Benutzeroberfläche von MapForce in ein Fenster Ihrer Applikation einbetten. Da Sie die gesamte Benutzeroberfläche von MapForce 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 MapForce 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 [MapForceControl](#). 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 [MapForceControlDocument](#) oder [MapForceControlPlaceHolder](#) ActiveX Controls nicht instantiiert oder aufgerufen werden.

Wenn Sie Initialisierungen vornehmen müssen oder ein bestimmtes Verhalten von MapForce automatisieren wollen, verwenden Sie die für [MapForceControl](#) beschriebenen Eigenschaften, Methoden und Events. Um komplexere MapForce Funktionsaufrufe auszuführen, sollten Sie eventuell [MapForceControl.Application](#) verwenden.

In diesem Abschnitt finden Sie ein Beispiel ([Beispiel: HTML](#)), in dem gezeigt wird, wie die MapForce 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.

### 33.1.1 Beispiel: HTML

In diesem Beispiel wird eine einfache Integration des MapForce Control auf Applikationsebene in eine HTML-Seite beschrieben. Die Integration wird in den folgenden Abschnitten beschrieben:

- Instantiieren Sie ein MapForceControl 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 MapForce Installation im folgenden Ordner: MapForceExamples\ActiveX\HTML\MapForceActiveX\_ApplicationLevel.htm.

**Anmerkung:** Dieses Beispiel funktioniert nur in Internet Explorer.

#### Instantiieren des Control

Der HTML Object Tag dient zum Erstellen einer Instanz des MapForceControl. Die Classid ist die von MapForceControl. Mit "width" und "height" wird die Fenstergröße definiert. Es sind keine weiteren Parameter erforderlich, da die Standardeinstellung "application-level" ist.

```
<OBJECT id="objMapForceControl"
 Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
 width="800"
 height="500"
 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 MapForceControl geöffnet. Wir verwenden eine Methode, um die Datei relativ zum MapForceControl anzugeben, daher kann das Beispiel in unterschiedlichen Installationen ausgeführt werden.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// -----
// open a pre-defined document
function BtnOpenMEFile()
{
 objMapForceControl.Open("C:\Documents and Settings\username\My
Documents\Altova\XMLSpy2012\Examples\MarketingExpenses.mfd");
}
</SCRIPT>
```

#### Hinzufügen von Schaltflächen für die Codegenerierung

Zwar stehen die Codegenerierungsfunktionen für das aktive Dokument über Menüs zur Verfügung, doch möchten wir Schaltflächen hinzufügen, mit denen Sie Code generieren können, ohne dass Sie vorher nach dem Pfad für die Ausgabe gefragt werden. Wir verwenden dazu eine ähnliche Methode wie im vorigen Abschnitt.



Zuerst kommen die Schaltflächen:

```
<input type="button" value="Generate XSLT" onclick="BtnGenerate(0)">
<input type="button" value="Generate Java" onclick="BtnGenerate(1)">
<input type="button" value="Generate C++" onclick="BtnGenerate(2)">
<input type="button" value="Generate C#" onclick="BtnGenerate(3)">
```

Anschließend kommt das Skript, mit dem der Code in Unterordnern des aktuell definierten Standardausgabeordners generiert wird.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// -----
// generate code for active document into language-specific sub folders of
// the current default output directory. No user interaction necessary.
function BtnGenerate(languageID)
{
 // get top-level object of automation interface
 var objApp = objMapForceControl.Application;

 // get the active document
 var objDocument = objApp.ActiveDocument;

 // retrieve object to set the generation output path
 var objOptions = objApp.Options;

 if (objDocument == null)
 alert("no active document found");
 else
 {
 if (languageID == 0)
 {
 objOptions.XSLTDefaultOutputDirectory =
objOptions.XSLTDefaultOutputDirectory + "\\XSLTGen";
 objDocument .GenerateXSLT();
 }
 else if (languageID == 1)
 {
 objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/JavaCode";
 objDocument .GenerateJavaCode();
 }
 else if (languageID == 2)
 {
 objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CPPCode";
 objDocument .GenerateCppCode();
 }
 else if (languageID == 3)
 {
 objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CSharpCode";
 objDocument .GenerateCHashCode();
 }
 }
}
</SCRIPT>
```

### Verbindung zu benutzerdefinierten Events herstellen

In diesem Beispiel werden zur Veranschaulichung des Prinzips zwei Event Callbacks für benutzerdefinierte MapForceControl Events implementiert:

```
<!-- ----- -->
```

```
<!-- custom event 'OnDocumentOpened' of MapForceControl object -->
<SCRIPT LANGUAGE="javascript">
 function objMapForceControl::OnDocumentOpened(objDocument)
 {
 // alert("Document '" + objDocument.Name + "' opened!");
 }
</SCRIPT>

<!-- ----- -->
<!-- custom event 'OnDocumentClosed' of MapForceControl object -->
<SCRIPT LANGUAGE="javascript">
 function objMapForceControl::OnDocumentClosed(objDocument)
 {
 // alert("Document '" + objDocument.Name + "' closed!");
 }
</SCRIPT>
```

## 33.2 Integration auf Dokumentebene

Bei der Integration auf Dokumentebene haben Sie Freiheit bei der Instantiierung und Platzierung der folgenden Teile der MapForce Benutzeroberfläche:

- Bearbeitungsfenster für MapForce Mappings
- MapForce Übersichtsfenster
- MapForce Bibliotheksfenster
- MapForce Validierungsfenster
- MapForce Projektfenster

Falls erforderlich, muss in Ihrer Applikation ein Ersatz für die Menüs und Symbolleisten von MapForce 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 MapForceControl OCX-Datei enthalten.

- [Mit Hilfe von MapForceControl](#) können Sie die Integrationsebene definieren und auf applikationsweite Funktionalitäten zugreifen.
- [Mit Hilfe von MapForceControlDocument](#) können Sie beliebig viele Editor-Fenster erstellen. Eventuell genügt es, nur ein Fenster zu erstellen und dieses wiederzuverwenden.
- Verwenden Sie optional [MapForceControlPlaceholder](#), um die MapForce Fenster "Übersicht", "Validierung" und "Projekt" einzubetten.
- Rufen Sie Laufzeitinformationen über in MapForceControl 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 MapForce Befehlen](#).

Wenn Sie ein bestimmtes Verhalten von MapForce automatisieren möchten, verwenden Sie die für [MapForceControl](#), [MapForceControlDocument](#) und [MapForceControlPlaceholder](#) beschriebenen Eigenschaften, Methoden und Events. Bei Aufruf komplexerer MapForce Funktionalitäten sollten Sie eventuell [MapForceControl.Application](#), [MapForceControlDocument.Document](#) und [MapForceControlPlaceholder.Project](#) verwenden. Verwenden Sie allerdings zum Öffnen eines Dokuments immer [MapForceControlDocument.Open](#) oder [MapForceControlDocument.New](#) am entsprechenden Document Control. Verwenden Sie zum Öffnen eines Projekts immer [MapForceControlPlaceholder.OpenProject](#) an einem Placeholder Control, das ein MapForce Projektfenster einbettet.

Beschreibungen zum Instantiieren und Aufrufen der erforderlichen Controls in verschiedenen Programmierungsumgebungen finden Sie unter [Beispiele](#).

### MapForce Integration und Deployment auf Client-Rechnern:

Wenn Sie eine .NET-Applikation erstellen und diese auf anderen Clients zur Verfügung stellen möchten, müssen Sie auf den Client Rechnern die folgenden Komponenten installieren:

- die MapForce Applikation
- das Integrationspaket für MapForce
- den benutzerdefinierten Integrationscode.

### 33.2.1 Verwendung von MapForceControl

Zur Integration auf Dokumentebene instantiiieren Sie zuerst ein [MapForceControl](#). 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 [MapForceControlDocument](#) und [MapForceControlPlaceholder](#).

Eine Beschreibung zur Integration von MapForce Befehlen in Ihre Applikation finden Sie unter [Abfragen von MapForce Befehlen](#). Senden Sie Befehle über die Methode [Exec](#) an MapForce. Führen Sie mit Hilfe der Methode [QueryStatus](#) eine Abfrage durch, um zu sehen, ob ein Befehl derzeit aktiviert oder deaktiviert ist.

### 33.2.2 Verwendung von MapForceControlDocument

Mit Hilfe einer Instanz des MapForceControlDocument ActiveX Control können Sie ein MapForce 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.

### 33.2.3 Verwendung von MapForceControlPlaceHolder

Mit Hilfe von Instanzen des MapForceControlPlaceHolder ActiveX Control können Sie selektiv weitere Eingabehilfefenster von MapForce in Ihre Applikation einbetten. Die Eigenschaft [PlaceholderWindowID](#) wählt das MapForce Eingabehilfefenster aus, um es einzubetten. Verwenden Sie für jeden Fenster-Identifizier nur einen MapForceControlPlaceHolder. Eine Beschreibung gültiger Fenster-Identifizier finden Sie unter .

Für Platzhalter-Controls, die das MapForce Projektfenster auswählen, stehen zusätzliche Methoden zur Verfügung. Verwenden Sie [OpenProject](#), um ein MapForce Projekt zu laden. Verwenden Sie die Eigenschaft [Project](#) und die Methoden und Eigenschaften aus dem MapForce Automation Interface, um andere projektbezogene Operationen auszuführen.

### 33.2.4 Abfragen von MapForce Befehlen

Bei der Integration auf Dokumentenebene stehen Ihrer Applikation kein Menü und keine Symbolleiste aus MapForce 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 MapForce stehen Ihnen sogar Bilder für in MapForce verwendete Befehle zur Verfügung. Symbole im GIF-Format finden Sie in Ihrem MapForce Installationsordner unter `MapForceExamples\ActiveX\Images of your MapForce`. Die Dateinamen entsprechen den [labels](#) von Befehlen.

### 33.2.5 Beispiele

Dieser Abschnitt enthält Beispiele für die Integration von MapForce auf Dokumentebene über unterschiedliche Container-Umgebungen und Programmiersprachen. Der Quellcode für alle Beispiele steht im Ordner `MapForceExamples\ActiveX` Ihrer MapForce Installation zur Verfügung.

#### C#

Im C#-Beispiel wird gezeigt, wie man das MapForceControl in eine allgemeine Desktop-Applikation integriert, die mit C# über Visual Studio 2008 erstellt wurde.

- Es werden die folgenden Themen behandelt:
  - Erstellen einer dynamischen Menüleiste auf Basis der Informationen aus der MapForceControl API.
  - Verwendung von des MapForce Platzhalter Control in einem Standard-Frame-Fenster.
  - Verwendung eines MapForce Platzhalter Control in einem in der Größe anpassbaren Tool-Fenster.
  - Wie man ein Event behandelt, das von der MapForceControl API ausgelöst wird.

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 MapForce -Komponenten zur Toolbox

Bevor Sie einen Blick in das Beispielpunkt werfen, fügen Sie bitte die Assemblies zur .NET IDE Toolbox hinzu. Der MapForce 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 `AxMapForceControl`, `AxMapForceControlDocument` und `AxMapForceControlPlaceholder` auf dem Register ".NET Framework Components" angezeigt. Aktivieren Sie alle, damit Sie in der IDE zur Verfügung stehen.

Nun können Sie die `MapForceApplication.sln` im Ordner `ActiveX\C#\MapForceApplication` öffnen, um das Projekt zu laden.

Nun können Sie die Datei `StyleVision.sln` im Ordner `MapForceExamples\ActiveX\C#\StyleVision` öffnen, um das Projekt zu laden.

#### Platzieren des MapForceControl

Sie benötigen eine MapForceControl Instanz, um die Integrationsebene zu definieren und die Dokument- und Platzhalter-Controls der MapForce 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 Beispielpunkt 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 MapForceControl. Die Eigenschaft



IntegrationLevel muss unbedingt auf **ICActiveXIntegrationOnDocumentLevel** gesetzt werden, um die Unterstützung der MapForce Bibliothek für Dokumente und Platzhalter zu aktivieren. Die Eigenschaften der in das MDIFrame-Fenster der Beispielapplikation platzierten MapForceControl Komponente werden unten gezeigt.

|                       |                                            |
|-----------------------|--------------------------------------------|
| ⊕ (DataBindings)      |                                            |
| ⊕ (DynamicProperties) |                                            |
| (Name)                | <b>axMapForceControl</b>                   |
| AccessibleDescription |                                            |
| AccessibleName        |                                            |
| AccessibleRole        | Default                                    |
| AllowDrop             | False                                      |
| Anchor                | Top, Left                                  |
| CausesValidation      | True                                       |
| Dock                  | None                                       |
| ImeMode               | NoControl                                  |
| IntegrationLevel      | <b>ICActiveXIntegrationOnDocumentLevel</b> |
| ⊕ Location            | <b>280; 8</b>                              |
| Locked                | False                                      |
| Modifiers             | Private                                    |
| ReadOnly              | <b>True</b>                                |
| ⊕ Size                | <b>224; 112</b>                            |
| TabIndex              | <b>1</b>                                   |
| TabStop               | <b>False</b>                               |
| Tag                   |                                            |
| Visible               | <b>False</b>                               |

Setzen Sie das Visible-Flag auf "False", damit es zu keinen Verwechslungen beim Control kommt.

Hinzufügen des Placeholder Control

### Placeholders im MDI-Frame

Im Beispielprojekt müssen Placeholder Controls in das MDI-Haupt-Frame eingebettet werden. Sie werden ebenfalls 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.

Unten sehen Sie die Eigenschaften des Bibliotheksfensters auf der linken Seite des MDIMain Frame-Fensters:

|   |                       |                                 |
|---|-----------------------|---------------------------------|
| + | (DataBindings)        |                                 |
| + | (DynamicProperties)   |                                 |
|   | (Name)                | <b>axMapForceControlLibrary</b> |
|   | AccessibleDescription |                                 |
|   | AccessibleName        |                                 |
|   | AccessibleRole        | Default                         |
|   | AllowDrop             | False                           |
|   | Anchor                | Top, Left                       |
|   | CausesValidation      | True                            |
|   | Dock                  | <b>Left</b>                     |
|   | ImeMode               | NoControl                       |
| + | Location              | <b>0; 0</b>                     |
|   | Locked                | False                           |
|   | Modifiers             | Private                         |
|   | PlaceholderWindowID   | <b>MapForceXLibraryWindow</b>   |
| + | Size                  | <b>272; 625</b>                 |
|   | TabIndex              | <b>2</b>                        |
|   | TabStop               | True                            |
|   | Tag                   |                                 |
|   | Visible               | True                            |

Im Folgenden sehen Sie die Eigenschaften des Ausgabefensters:

|   |                       |                                  |
|---|-----------------------|----------------------------------|
| + | (DataBindings)        |                                  |
| + | (DynamicProperties)   |                                  |
|   | (Name)                | <b>axMapForceControlOutput</b>   |
|   | AccessibleDescription |                                  |
|   | AccessibleName        |                                  |
|   | AccessibleRole        | Default                          |
|   | AllowDrop             | False                            |
|   | Anchor                | Top, Left                        |
|   | CausesValidation      | True                             |
|   | Dock                  | <b>Bottom</b>                    |
|   | ImeMode               | NoControl                        |
| + | Location              | <b>272; 473</b>                  |
|   | Locked                | False                            |
|   | Modifiers             | Private                          |
|   | PlaceholderWindowID   | <b>MapForceXValidationWindow</b> |
| + | Size                  | <b>620; 152</b>                  |
|   | TabIndex              | <b>4</b>                         |
|   | TabStop               | True                             |
|   | Tag                   |                                  |
|   | Visible               | True                             |

Für die Placeholder wurden auch die Eigenschaften "Anchor" und "Dock" definiert, um auf eine Anpassung der Größe des Frame-Fensters zu reagieren.

### Placeholder in einem separaten Tool-Fenster

Das Placeholder Control kann auch in ein separates frei schwebendes Tool-Fenster platziert werden. Erstellen Sie dazu ein neues Formular als Tool-Fenster und fügen Sie das Control wie

oben gezeigt hinzu. Das MapForceOverviewWnd im Beispielprojekt enthält das Übersichtsfenster von MapForce.

Eigenschaften im Tool-Fenster "Übersicht":

|   |                       |                                  |
|---|-----------------------|----------------------------------|
| + | (DataBindings)        |                                  |
| + | (DynamicProperties)   |                                  |
|   | (Name)                | <b>axMapForceControlOverview</b> |
|   | AccessibleDescription |                                  |
|   | AccessibleName        |                                  |
|   | AccessibleRole        | Default                          |
|   | AllowDrop             | False                            |
|   | Anchor                | <b>Top, Bottom, Left, Right</b>  |
|   | CausesValidation      | True                             |
|   | Dock                  | None                             |
|   | ImeMode               | NoControl                        |
| + | Location              | <b>0; 0</b>                      |
|   | Locked                | False                            |
|   | Modifiers             | <b>Public</b>                    |
|   | PlaceholderWindowID   | <b>MapForceXOverviewWindow</b>   |
| + | Size                  | <b>292; 266</b>                  |
|   | TabIndex              | <b>0</b>                         |
|   | TabStop               | True                             |
|   | Tag                   |                                  |
|   | Visible               | True                             |

Alle Placeholder Controls benötigen jedoch eine Verbindung zum Haupt-MapForceControl. Normalerweise kann diese Verbindung automatisch hergestellt werden, sodass nichts mehr zu tun ist. Die beiden Placeholder im MDI-Frame funktionieren so. Im Falle des Placeholder Control im Tool-Fenster müssen wir in MDIMain.cs Code zur public MDIMain() Methode hinzufügen:

```
m_MapForceOverview = new MapForceOverviewWnd();

MapForceControlLib.MapForceControlPlaceholderClass type =
(MapForceControlLib.MapForceControlPlaceholderClass)m_MapForceOverview.axMapForceControlOverview.GetOcx();
type.AssignMultiDocCtrl((MapForceControlLib.MapForceControlClass)axMapForceControl.GetOcx());

m_MapForceOverview.Show();
```

Hier wird das MapForceOverviewWnd erstellt und gezeigt. Zusätzlich dazu wird eine spezielle Methode des Placeholder Control aufgerufen, um das MapForce damit zu verbinden. AssignMultiDocCtrl() erhält das MapForceControl als Parameter und registriert eine Referenz darauf im Placeholder Control.

#### Abrufen von Befehlsinformationen

Über das MapForceControl haben Sie über die Eigenschaft CommandsStructure Zugriff auf alle Befehle von MapForce. Im Beispielprojekt werden die Schnittstellen [MapForceCommands](#) und [MapForceCommand](#) 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 MapForceApplication Klasse platziert:

```

public MDIMain()
{
 .
 .
 .
 MapForceControlLib.MapForceCommands objCommands;
 objCommands = axMapForceControl.CommandsStructure;

 long nCount = objCommands.Count;

 for(long idx = 0;idx < nCount;idx++)
 {
 MapForceControlLib.MapForceCommand objCommand;
 objCommand = objCommands[(int)idx];

 // We are looking for the Menu with the name IDR_MAPFORCE. This menu
 // contains
 // the complete main menu of MapForce.

 if(objCommand.Label == "IDR_MAPFORCE")
 {
 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 MapForce Menü aus dem IDR\_MAPFORCE Befehlsobjekt und fügt die MapForce 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 MapForce 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_MapForceCmdID = (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("MapForce
 Application.CustomMenuItem"))
 {
 CustomMenuItem customItem = (CustomMenuItem)sender;
 }
}

```

```

 ProcessCommand(customItem.m_MapForceCmdID);
 }
}

private void ProcessCommand(int nID)
{
 MapForceDoc docMapForce = GetCurrentMapForceDoc();

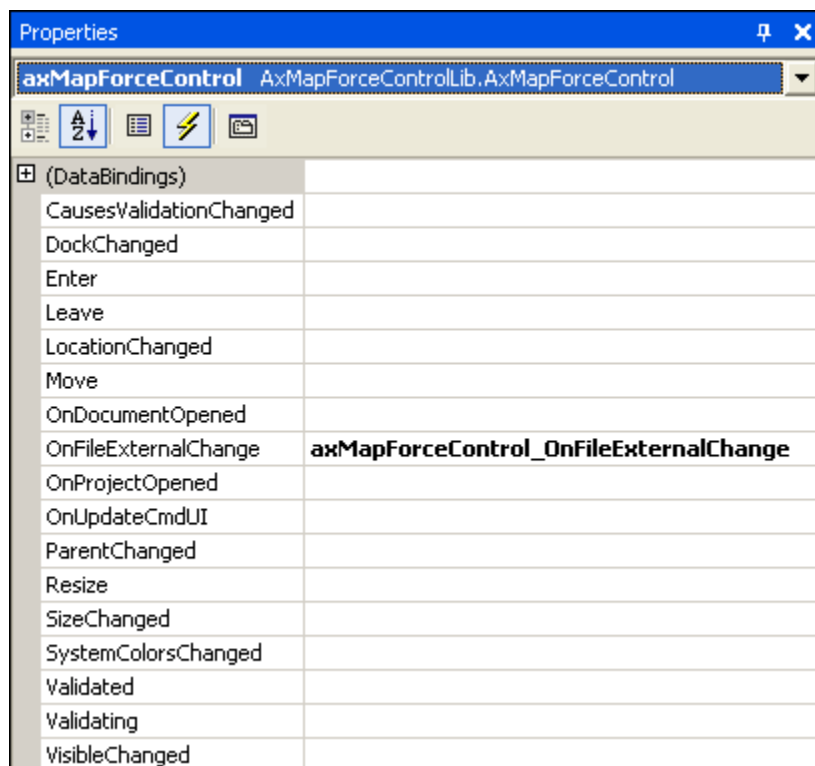
 if(docMapForce != null)
 docMapForce.axMapForceControlDoc.Exec(nID);
 else
 axMapForceControl.Exec(nID);
}

```

ProcessCommand delegiert die Ausführung entweder an das MapForceControl selbst oder an ein aktives MapForce Dokument, das in einem MapForceControlDocument Control geladen ist. Dies ist notwendig, da das MapForceControl nicht wissen kann, welches Dokument gerade in der Host-Applikation aktiv ist.

#### Behandlung von Events

Da alle Events in der MapForce Bibliothek auf Connection Points basieren, können Sie mit Hilfe der C# Delegatsfunktionalität benutzerdefinierte Event Handler bereitstellen. Auf der Seite "Eigenschaften" der einzelnen Controls der MapForce Bibliothek finden Sie jeweils eine vollständige Liste von Events. In der Abbildung unten sehen Sie die Events des Haupt-MapForceControl:



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

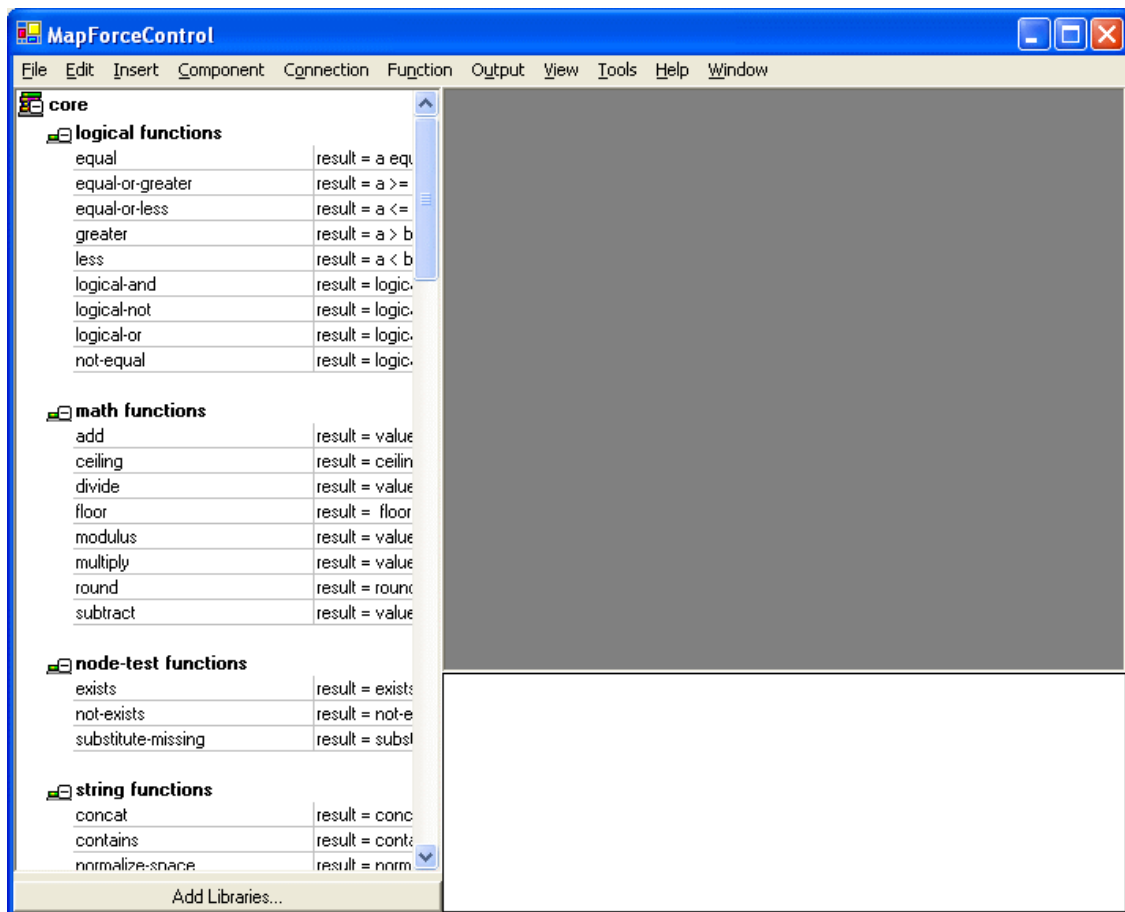
und zeigt ein Meldungsfeld an, um den Benutzer zu informieren, dass eine vom MapForceControl geladene Datei von außerhalb geändert wurde:

```
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!");

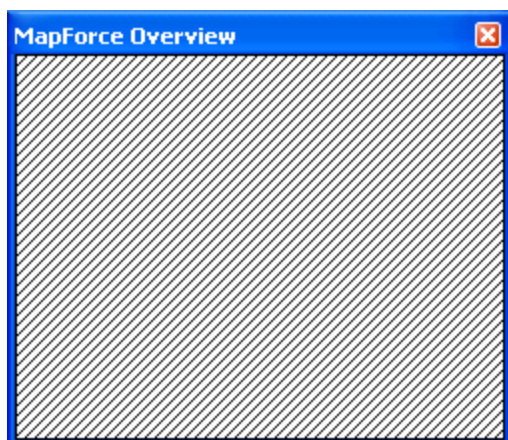
 // This turns off any file reloading:
 e.varRet = false;
}
```

### Testen des Beispiels

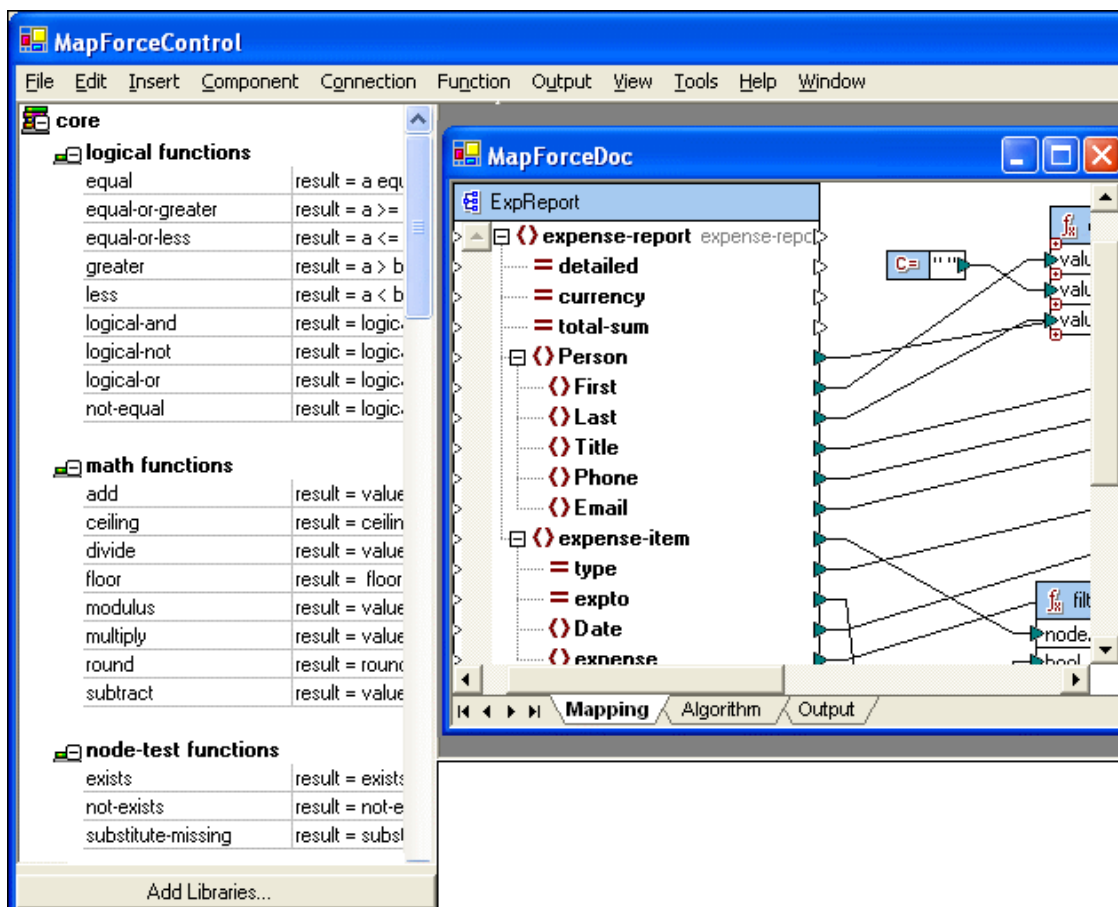
Nachdem Sie die Assemblies zur Toolbox hinzugefügt haben (siehe [Einführung](#)), können Sie das Beispielprojekt mit F5 ausführen, ohne irgendwelche Änderungen im Code vornehmen zu müssen. Das MDI-Haupt-Frame-Fenster wird zusammen mit einem frei schwebenden Tool-Fenster erstellt, das das Übersichtsfenster von MapForce enthält. Die Applikation sieht in etwa so aus, wie in der Abbildung unten:



Auch das frei schwebende Übersichts-Tool-Fenster wird erstellt:



Wählen Sie **Datei | Öffnen** um die Datei `MarketingExpenses.mfd` aus dem MapForce examples-Ordner zu öffnen. Die Datei wird geladen und in einem eigenen Dokument-Child-Fenster angezeigt:



Nachdem Sie das Dokument geladen haben, können Sie versuchen, Menübefehle zu verwenden. Beachten Sie, dass auch Kontextmenüs zur Verfügung stehen. Wahlweise können auch zusätzliche Dokumente geladen werden. Speichern Sie Ihre Änderungen mit dem Befehl **Datei | Speichern**.

## HTML

In diesem Beispiel wird eine Integration des MapForce Control auf Dokumentenebene in eine HTML-Seite gezeigt. Es werden die folgenden Themen behandelt:

- Instantiieren eines MapForceControl ActiveX Control-Objekts in HTML-Code
- Instantiieren eines MapForceControlDocument ActiveX Control für die Bearbeitung eines MapForce Mappings
- Instantiieren eines MapForceControlPlaceholder für ein MapForce-Projektfenster
- Instantiieren eines MapForceControlPlaceholder ActiveX Control für die Aufnahme eines der MapForce-Eingabehilfefenster
- Erstellen einer benutzerdefinierten Symbolleiste für einige häufig gebrauchte MapForce-Befehle
- Hinzufügen von weiteren Schaltflächen, die das COM Automation Interface von MapForce verwenden
- Verwenden von Event Handlern zum Aktualisieren von Befehlsschaltflächen

Dieses Beispiel ist vollständig in der Datei MapForceActiveX\_ApplicationLevel.htm im Ordner C:\Program Files\Altova\MapForce2012\Examples\ActiveX\HTML\ Ihrer MapForce Installation enthalten.

**Anmerkung:** Dieses Beispiel funktioniert nur in Internet Explorer.

### Instantiieren des MapForceControl

Zum Erstellen einer Instanz des MapForceControl wird der HTML OBJECT Tag verwendet. Die Classid ist die von MapForceControl. "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="objMapForceXMapForceControl"
 Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
 width="0"
 height="0"
 VIEWASTEXT>
 <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

### Erstellen eines Bearbeitungsfensters

Der HTML OBJECT Tag wird zum Einbetten eines Dokumentenbearbeitungsfensters verwendet. Der zusätzliche benutzerdefinierte Parameter definiert, dass das Control mit einem neuen leeren Mapping initialisiert werden muss.

```
<OBJECT id="objDoc1"
 Classid="clsid:DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
 width="600"
 height="500"
 VIEWASTEXT>
 <PARAM NAME="NewDocument">
</OBJECT>
```

### Erstellen eines Projektfensters

Der HTML OBJECT Tag wird zum Erstellen eines MapForceControlPlaceholder-Fensters verwendet. Der erste zusätzliche benutzerdefinierte Parameter definiert, welcher Platzhalter das MapForce-Projektfenster anzeigen soll. Der zweite Parameter lädt eines der Beispielprojekte aus Ihrer MapForce 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="MapForceExamples/MapForceExamples.mfp">
</OBJECT>

```

#### Erstellen eines Placeholder für Eingabehilfenfenster

Der HTML OBJECT Tag wird zum Instantiieren eines MapForceControlPlaceHolder ActiveX Control verwendet, das die verschiedenen MapForce Hilfsfenster aufnehmen kann. Anfangs wird kein Hilfsfenster angezeigt. Siehe Beispieldatei:

```

<OBJECT id="objEHWindow"
 Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
 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 definierten Wert gesetzt.

```

<input type="button" value="Library Window" onclick="BtnHelperWindow(0)">
<input type="button" value="Overview Window" onclick="BtnHelperWindow(1)">
<input type="button" value="Validation Window" 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 MapForce Befehle. Sie finden die Befehls-ID-Nummern in den Schaltflächen-Elementen unter

```

<button id="btnInsertXML" title="Insert XML Schema/File"
onclick="BtnDoCommand(13635)">

</button>
<button id="btnInsertDB" title="Insert Database"
onclick="BtnDoCommand(13590)">

</button>
<button id="btnInsertEDI" title="Insert EDI" onclick="BtnDoCommand(13591)">

</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)
{
 objMapForceX.Exec(cmdID);
 msgtext.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 MarketingExpenses"
onclick="BtnOpenMEFile(objDoc1)">
</p>
<p>
 <input type="button" id="GenerateXSLT" value="Generate XSLT"
onclick="BtnGenerate(objDoc1, 0)">
 <input type="button" id="GenerateJava" value="Generate Java"
onclick="BtnGenerate(objDoc1, 1)">
 <input type="button" id="GenerateCpp" value="Generate C++"
onclick="BtnGenerate(objDoc1, 2)">
 <input type="button" id="GenerateCSharp" value="Generate C#"
onclick="BtnGenerate(objDoc1, 3)">
</p>
```

Das entsprechende JavaScript sieht folgendermaßen aus:

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// -----
// open a document in the specified document control window.
function BtnOpenMEFile(objDocCtrl)
{
 // do not use MapForceX.Application.OpenDocument(...) to open a
document,
 // since then MapForceControl wouldn't know a control window to show
 // the document in. Instead:

 objDocCtrl.OpenDocument("C:\Documents and Settings\username\My
Documents\
 Altova\XMLSpy2009\Examples/MarketingExpenses.mfd");
 objDocCtrl.setActive();
}

// -----
// open a new empty document in the specified document control window.
function BtnNewFile(objDocCtrl)
{
 objDocCtrl.OpenDocument("");
 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.SaveDocument();
 }
 else
 {
 alert("Please set path for the document first!");
 strPath.focus();
 }
 }

 objDocCtrl.setActive();
}
</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 `MapForceControl` gibt uns die Gelegenheit, den Schaltflächenstatus "aktiviert/deaktiviert", der mit `MapForce` Befehlen verknüpft ist, zu aktualisieren. Um abzufragen, ob ein Befehl aktiv oder deaktiviert ist, wird die Methode [MapForceControl.QueryStatus](#) verwendet.

```

<SCRIPT LANGUAGE="javascript">

function objMapForceX::OnUpdateCmdUI()
{
 if (document.readyState == "complete") // 'complete'
 {
 // update status of buttons
 GenerateXSLT.disabled = ! (objDoc1.QueryStatus(13617) & 0x02);
 // not enabled
 GenerateJava.disabled = ! (objDoc1.QueryStatus(13587) & 0x02);
 // not enabled
 GenerateCpp.disabled = ! (objDoc1.QueryStatus(13589) & 0x02);
 // not enabled
 GenerateCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);
 // not enabled

 btnFuncUserDef.disabled = ! (objDoc1.QueryStatus(13633) & 0x02);
 btnFuncUserDefSel.disabled = ! (objDoc1.QueryStatus(13634) &
0x02);
 btnFuncSettings.disabled = ! (objDoc1.QueryStatus(13632) & 0x02
);
 btnInsertInput.disabled = ! (objDoc1.QueryStatus(13491) & 0x02);

 btnGenXSLT.disabled = ! (objDoc1.QueryStatus(13617) & 0x02);
 btnGenXSLT2.disabled = ! (objDoc1.QueryStatus(13618) & 0x02);
 btnGenXQuery.disabled = ! (objDoc1.QueryStatus(13586) & 0x02);
 btnGenCPP.disabled = ! (objDoc1.QueryStatus(13589) & 0x02);
 btnGenCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);
 btnGenJava.disabled = ! (objDoc1.QueryStatus(13587) & 0x02);
 }
}

```

```
// set activity status of simulated toolbar
</SCRIPT>
```

**Visual Basic**

Der Quellcode für eine Integration von MapForceControl in ein VisualBasic-Programm befindet sich im Ordner MapForceExamples\ActiveX\VisualBasic6 (relativ zu Ihrer MapForce Installation).

### 33.3 Befehlstabelle für MapForce

In den Tabellen in diesem Abschnitt werden die Namen und Identifier aller Befehle aufgelistet, die in MapForce zur Verfügung stehen. In den einzelnen Unterabschnitten werden jeweils die Befehle aus dem entsprechenden Menü von MapForce 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 MapForce Edition eventuell nicht unterstützt. Eine Beschreibung, wie man die aktuelle Ressourcenstruktur und die Verfügbarkeit von Befehlen abfragt, finden Sie unter [Abfragen von MapForce Befehlen](#).

Verwenden Sie die Befehls-Identifier mit [MapForceControl.QueryStatus](#) oder [MapForceControlDocument.QueryStatus](#) um den aktuellen Status eines Befehls zu überprüfen. Verwenden Sie [MapForceControl.Exec](#) oder [MapForceControlDocument.Exec](#) um einen Befehl auszuführen.

[Menü "Datei"](#)  
[Menü "Bearbeiten"](#)  
[Menü "Einfügen"](#)  
[Menü "Projekt"](#)  
[Menü "Komponente"](#)  
[Menü "Verbindung"](#)  
[Menü "Funktion"](#)  
[Menü "Ausgabe"](#)  
[Menü "Ansicht"](#)  
[Menü "Extras"](#)  
[Menü "Fenster"](#)  
[Menü "Hilfe"](#)

[Befehle, die sich nicht im Hauptmenü befinden](#)

### 33.3.1 Menü "Datei"

Befehle aus dem Menü "Datei":

| Menütext                                | Befehlsname                    | ID    |
|-----------------------------------------|--------------------------------|-------|
| Neu...                                  | ID_FILE_NEW                    | 57600 |
| Öffnen...                               | ID_FILE_OPEN                   | 57601 |
| Speichern                               | ID_FILE_SAVE                   | 57603 |
| Speichern unter...                      | ID_FILE_SAVE_AS                | 57604 |
| Alles speichern                         | ID_FILE_SAVEALL                | 32377 |
| Neu laden                               | IDC_FILE_RELOAD                | 32467 |
| Schließen                               | ID_WINDOW_CLOSE                | 32453 |
| Alles schließen                         | ID_WINDOW_CLOSEALL             | 32454 |
| Projekt speichern                       | ID_FILE_SAVEPROJECT            | 32378 |
| Projekt schließen                       | ID_FILE_CLOSEPROJECT           | 32355 |
| Drucken...                              | IDC_FILE_PRINT                 | 57607 |
| Druckvorschau                           | IDC_FILE_PRINT_PREVIEW         | 57609 |
| Druckereinrichtung...                   | ID_FILE_PRINT_SETUP            | 57606 |
| Mapping validieren                      | ID_MAPPING_VALIDATE            | 32347 |
| Code in ausgewählter Sprache generieren | ID_FILE_GENERATE_SELECTED_CODE | 32362 |
| Code generieren in/XSLT 1.0             | ID_FILE_GENERATEXSLT           | 32360 |
| Code generieren in/XSLT 2.0             | ID_FILE_GENERATEXSLT2          | 32361 |
| Code generieren in/XQuery               | ID_FILE_GENERATEXQUERY         | 32359 |
| Code generieren in/Java                 | ID_FILE_GENERATEJAVACODE       | 32358 |
| Code generieren in/C# (Sharp)           | ID_FILE_GENERATECSCODE         | 32357 |
| Code generieren in/C++                  | ID_FILE_GENERATECPPCODE        | 32356 |
| Dokumentation generieren...             | ID_FILE_GENERATE_DOCUMENTATION | 32468 |
| Mappingeinstellungen...                 | ID_MAPPING_SETTINGS            | 32396 |
| Letzte Dateien/Letzte Datei             | ID_FILE_MRU_FILE1              | 57616 |
| Letzte Projekte/Letztes Projekt         | ID_FILE_MRU_PROJECT1           | 32364 |
| Beenden                                 | ID_APP_EXIT                    | 57665 |

### 33.3.2 Menü "Bearbeiten"

Befehle aus dem Menü "Bearbeiten":

| Menütext          | Befehlsname        | ID    |
|-------------------|--------------------|-------|
| Rückgängig        | ID_EDIT_UNDO       | 57643 |
| Wiederherstellen  | ID_EDIT_REDO       | 57644 |
| Suchen...         | ID_EDIT_FIND       | 57636 |
| Weitersuchen      | ID_EDIT_FINDNEXT   | 32349 |
| Vorheriges suchen | ID_EDIT_FINDPREV   | 32350 |
| Ausschneiden      | ID_EDIT_CUT        | 57635 |
| Kopieren          | ID_EDIT_COPY       | 57634 |
| Einfügen          | ID_EDIT_PASTE      | 57637 |
| Löschen           | ID_EDIT_CLEAR      | 57632 |
| Alle auswählen    | ID_EDIT_SELECT_ALL | 57642 |

### 33.3.3 Menü "Einfügen"

Befehle aus dem Menü "Einfügen":

| Menütext             | Befehlsname                   | ID    |
|----------------------|-------------------------------|-------|
| XML-Schema/Datei     | ID_INSERT_XSD                 | 32393 |
| Datenbank            | ID_INSERT_DATABASE            | 32389 |
| EDI                  | ID_INSERT_EDI                 | 32390 |
| Textdatei            | ID_INSERT_TXT                 | 32392 |
| Webservice-Funktion  | ID_INSERT_WEBSERVICE_FUNCTION | 32319 |
| Excel 2007-Datei...  | ID_INSERT_EXCEL               | 32376 |
| XBRL-Dokument...     | ID_INSERT_XBRL                | 32469 |
| Konstante            | ID_INSERT_CONSTANT            | 32388 |
| Filter: Nodes/Zeilen | ID_INSERT_FILTER              | 32391 |
| SQL-WHERE-Bedingung  | ID_INSERT_SQLWHERE_CONDITION  | 32351 |
| Wertezuordnung       | ID_INSERT_VALUEMAP            | 32354 |
| IF-Else-Bedingung    | ID_INSERT_CONDITION           | 32394 |
| Ausnahme             | ID_INSERT_EXCEPTION           | 32311 |



### 33.3.4 Menü "Projekt"

Befehle aus dem Menü "Projekt":

| Menütext                                  | Befehlsname                        | ID    |
|-------------------------------------------|------------------------------------|-------|
| Dateien zu Projekt hinzufügen...          | ID_PROJECT_ADDFILESTOPROJECT       | 32420 |
| Aktive Datei zu Projekt hinzufügen...     | ID_PROJECT_ADDACTIVEFILETOPROJECT  | 32419 |
| Ordner erstellen                          | ID_POPUP_PROJECT_CREATE_FOLDER     | 32310 |
| Mapping öffnen                            | ID_POPUP_OPENOPERATIONSMAPPING     | 13692 |
| Mapping für Operation erstellen...        | ID_POPUP_CREATEMAPPINGFOROPERATION | 32399 |
| Mapping-Datei für Operation hinzufügen... | ID_POPUP_PROJECT_ADD_MAPPING       | 32309 |
| Projekt neu laden                         | ID_PROJECT_RELOAD                  | 32374 |
| Webservice einfügen...                    | ID_POPUP_PROJECT_INSERT_WEBSERVICE | 32306 |
| Datei in XMLSpy öffnen                    | ID_POPUP_PROJECT_OPENINXMLSPY      | 32305 |
| Code für das gesamte Projekt generieren   | ID_POPUP_PROJECT_GENERATE_PROJECT  | 32304 |
| Code generieren in/XSLT 1.0               | ID_PROJECT_GENERATEXSLT            | 32425 |
| Code generieren in/XSLT 2.0               | ID_PROJECT_GENERATEXSLT2           | 32426 |
| Code generieren in/XQuery                 | ID_PROJECT_GENERATEXQUERY          | 32424 |
| Code generieren in/Java                   | ID_PROJECT_GENERATEJAVACODE        | 32423 |
| Code generieren in/C# (Sharp)             | ID_PROJECT_GENERATECSCODE          | 32422 |
| Code generieren in/C++                    | ID_PROJECT_GENERATECPPCODE         | 32421 |
| Projekteigenschaften...                   | ID_PROJECT_PROPERTIES              | 32404 |

### 33.3.5 Menü "Komponente"

Befehle aus dem Menü "Komponente":

| Menütext                               | Befehlsname                        | ID    |
|----------------------------------------|------------------------------------|-------|
| Links ausrichten                       | ID_COMPONENT_LEFTALIGNTREE         | 32338 |
| Rechts ausrichten                      | ID_COMPONENT_RIGHTALIGNTREE        | 32340 |
| Root-Element ändern                    | ID_COMPONENT_CHANGEROOTELEMENT     | 32334 |
| Schema-Definition in XMLSpy bearbeiten | ID_COMPONENT_EDIT_SCHEMA           | 32337 |
| FlexText-Konfiguration bearbeiten      | ID_COMPONENT_EDIT_MFT              | 32301 |
| Tabellen hinzufügen/entfernen...       | ID_COMPONENT_SELECTTABLES          | 32346 |
| Aktualisieren                          | IDC_COMMAND_REFRESH_COMPONENT      | 32373 |
| Duplikat erzeugen                      | ID_COMPONENT_CREATE_DUPLICATE_ICON | 32335 |
| Duplikat löschen                       | ID_COMPONENT_REMOVE_DUPLICATE_ICON | 32339 |
| Datenbank-Tabelle - Aktionen           | ID_POPUP_DATABASETABLEACTIONS      | 32400 |
| Datenbankschlüssel-Einstellungen       | ID_POPUP_VALUEKEYSETTINGS          | 32417 |
| Datenbank abfragen                     | ID_QUERY_DATABASE                  | 32341 |
| Eigenschaften                          | ID_COMPONENT_PROPERTIES            | 32336 |

### 33.3.6 Menü "Verbindung"

Befehle aus dem Menü "Verbindung":

| Menütext                                             | Befehlsname                            | ID    |
|------------------------------------------------------|----------------------------------------|-------|
| Idente Sub-Einträge automatisch verbinden            | ID_CONNECTION_AUTOCONNECTCHILDREN      | 32342 |
| Einstellungen für "Idente Sub-Einträge verbinden"... | ID_CONNECTION_SETTINGS                 | 32344 |
| Idente Sub-Einträge verbinden                        | ID_CONNECTION_MAPCHILDELEMENTS         | 32343 |
| Zielorientiert (Standard)                            | ID_POPUP_NORMALCONNECTION              | 32401 |
| Alles kopieren (Sub-Einträge kopieren)               | ID_POPUP_NORMALWITHCHILDREN_CONNECTION | 32460 |
| Quellorientiertes Mapping (mixed-content)            | ID_POPUP_ORDERBYSOURCECONNECTION       | 32403 |
| Eigenschaften                                        | ID_POPUP_CONNECTION_SETTINGS           | 32398 |

### 33.3.7 Menü "Funktion"

Befehle aus dem Menü "Funktion":

| Menütext                                             | Befehlsname                       | ID    |
|------------------------------------------------------|-----------------------------------|-------|
| Benutzerdefinierte Funktion erstellen...             | ID_FUNCTION_CREATE_EMPTY          | 32380 |
| Benutzerdefinierte Funktion von Auswahl erstellen... | ID_FUNCTION_CREATE_FROM_SELECTION | 32381 |
| Funktion - Einstellungen...                          | ID_FUNCTION_SETTINGS              | 32387 |
| Funktion entfernen                                   | ID_FUNCTION_REMOVE                | 32385 |
| Input-Komponente einfügen                            | ID_FUNCTION_INSERT_INPUT          | 32383 |
| Output-Komponente einfügen...                        | ID_FUNCTION_INSERT_OUTPUT         | 32402 |

### 33.3.8 Menü "Ausgabe"

Befehle aus dem Menü "Ausgabe":

| Menütext                      | Befehlsname                     | ID    |
|-------------------------------|---------------------------------|-------|
| XSLT 1.0                      | ID_SELECT_LANGUAGE_XSLT         | 32433 |
| XSLT 2.0                      | ID_SELECT_LANGUAGE_XSLT2        | 32434 |
| XQuery                        | ID_SELECT_LANGUAGE_XQUERY       | 32432 |
| Java                          | ID_SELECT_LANGUAGE_JAVA         | 32431 |
| C# (Sharp)                    | ID_SELECT_LANGUAGE_CSHARP       | 32430 |
| C++                           | ID_SELECT_LANGUAGE_CPP          | 32429 |
| Ausgabedatei validieren       | ID_XML_VALIDATE                 | 32458 |
| Ausgabedatei speichern...     | IDC_FILE_SAVEGENERATEDOUTPUT    | 32321 |
| SQL-Script ausführen          | ID_TRANSFORM_RUN_SQL            | 32442 |
| Lesenzeichen einfügen/löschen | ID_TOGGLE_BOOKMARK              | 32317 |
| Nächstes Lesezeichen          | ID_GOTONEXTBOOKMARK             | 32315 |
| Vorhergehendes Lesezeichen    | ID_GOTOPREVBOKMARK              | 32314 |
| Alle Lesezeichen löschen      | ID_REMOVEALLBOOKMARKS           | 32313 |
| Pretty-Print                  | ID_PRETTY_PRINT_OUTPUT          | 32363 |
| Alle Ausgabedateien speichern | IDC_FILE_SAVEALLGENERATEDOUTPUT | 32374 |
| Ausgabedatei neu generieren   | ID_REGENERATE_PREVIEW_OUTPUT    | 32480 |
| Einstellungen für Textansicht | ID_TEXTVIEWSETTINGSDIALOG       | 32472 |

### 33.3.9 Menü "Ansicht"

Befehle aus dem Menü "Ansicht":

| Menütext                                    | Befehlsname                               | ID    |
|---------------------------------------------|-------------------------------------------|-------|
| Annotationen anzeigen                       | ID_SHOW_ANNOTATION                        | 32435 |
| Datentypen anzeigen                         | ID_SHOW_TYPES                             | 32437 |
| Bibliothek in Funktionstitelleiste anzeigen | ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER       | 32448 |
| Tipps anzeigen                              | ID_SHOW_TIPS                              | 32436 |
| Ausgewählte Komponentenkonnektoren anzeigen | ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS | 32443 |
| Quell- und Zielkonnektoren anzeigen         | ID_VIEW_RECURSIVEAUTOHIGHLIGHT            | 32447 |
| Vergrößern/Verkleinern                      | ID_VIEW_ZOOM                              | 32451 |
| Statusleiste                                | ID_VIEW_STATUS_BAR                        | 32449 |
| Bibliotheksfenster                          | ID_VIEW_LIBRARY_WINDOW                    | 32445 |
| Meldungen                                   | ID_VIEW_VALIDATION_OUTPUT                 | 32450 |
| Übersicht                                   | ID_VIEW_OVERVIEW_WINDOW                   | 32446 |
| Projektfenster                              | ID_VIEW_PROJECT_WINDOW                    | 32302 |
| XBRL-Anzeigeoptionen                        | ID_VIEW_XBRL_DISPLAY_OPTIONS              | 32473 |
| Zurück                                      | ID_CMD_BACK                               | 32479 |
| Vorwärts                                    | ID_CMD_FORWARD                            | 32478 |

### 33.3.10 Menü "Extras"

Befehle aus dem Menü "Extras":

| Menütext                                      | Befehlsname                       | ID    |
|-----------------------------------------------|-----------------------------------|-------|
| Globale Ressourcen                            | IDC_GLOBALRESOURCES               | 37401 |
| Aktive Konfiguration                          | IDC_GLOBALRESOURCES_SUBMENUENTRY1 | 37408 |
| Anpassen...                                   | ID_VIEW_CUSTOMIZE                 | 32444 |
| Optionen...                                   | ID_TOOLS_OPTIONS                  | 32441 |
| Symbolleisten und Fenster<br>wiederherstellen | ID_APP_RESET_TOOLBARS_AND_WINDOWS | 32956 |

### 33.3.11 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 |



### 33.3.12 Menü "Hilfe"

Befehle aus dem Menü "Hilfe":

| Menütext                       | Befehlsname                     | ID    |
|--------------------------------|---------------------------------|-------|
| Inhaltsverzeichnis...          | IDC_HELP_CONTENTS               | 32322 |
| Index..                        | IDC_HELP_INDEX                  | 32323 |
| Suchen...                      | IDC_HELP_SEARCH                 | 32324 |
| Software-Aktivierung...        | IDC_ACTIVATION                  | 32701 |
| Bestellformular                | IDC_ACTIVATION                  | 32326 |
| Registrieren...                | IDC_REGISTRATION                | 32330 |
| Auf Updates überprüfen...      | IDC_CHECK_FOR_UPDATES           | 32700 |
| Support Center...              | IDC_OPEN_SUPPORT_PAGE           | 32327 |
| Fragen und Antworten im Web... | IDC_SHOW_FAQ                    | 32331 |
| Komponenten downloaden...      | IDC_OPEN_COMPONENTS_PAGE        | 32325 |
| MapForce im Internet..         | IDC_OPEN_XML_SPY_HOME           | 32328 |
| MapForce Training...           | IDC_OPEN_MAPFORCE_TRAINING_PAGE | 32300 |
| Über MapForce...               | ID_APP_ABOUT                    | 57664 |

### 33.3.13 Befehle, die sich nicht im Hauptmenü befinden

Befehle, die sich nicht im Hauptmenü befinden:

| Menütext                                            | Befehlsname                             | ID    |
|-----------------------------------------------------|-----------------------------------------|-------|
|                                                     | IDC_QUICK_HELP                          | 32329 |
| FlexText-Konfiguration bearbeiten                   | ID_COMPONENT_EDIT_MFT                   | 32301 |
| Priorität - Kontext                                 | ID_COMPONENT_PRIORITYCONTEXT            | 32318 |
|                                                     | ID_EDIT_FINDPREV                        | 32350 |
|                                                     | ID_FUNCTION_GOTO_MAIN                   | 32382 |
| Input einfügen                                      | ID_FUNCTION_INSERT_INPUT_AT_POINT       | 32384 |
|                                                     | ID_FUNCTION_REMOVE                      | 32385 |
| Komponente durch interne Funktionsstruktur ersetzen | ID_FUNCTION_REPLACE_WITH_COMPONENTS     | 32386 |
|                                                     | ID_MAPFORCEVIEW_ZOOM                    | 32395 |
|                                                     | ID_NEXT_PANE                            | 32397 |
| Aktive Datei zum Projekt hinzufügen                 | ID_POPUP_PROJECT_ADDACTIVEFILETOPROJECT | 32405 |
| Dateien zu Projekt hinzufügen...                    | ID_POPUP_PROJECT_ADDFILESTOPROJECT      | 32406 |
| C++                                                 | ID_POPUP_PROJECT_GENERATECPPCODE        | 32408 |
| C# (Sharp)                                          | ID_POPUP_PROJECT_GENERATECSCODE         | 32409 |
| Java                                                | ID_POPUP_PROJECT_GENERATEJAVACODE       | 32410 |
| XQuery                                              | ID_POPUP_PROJECT_GENERATEXQUERY         | 32411 |
| XSLT 1.0                                            | ID_POPUP_PROJECT_GENERATEXSLT           | 32412 |
| XSLT 2.0                                            | ID_POPUP_PROJECT_GENERATEXSLT2          | 32413 |
| Alle generieren                                     | ID_POPUP_PROJECT_GENERATE_ALL           | 32303 |
| Code in Standardsprache generieren                  | ID_POPUP_PROJECT_GENERATE_CODE          | 32414 |
| Öffnen                                              | ID_POPUP_PROJECT_OPEN_MAPPING           | 32307 |
| Eigenschaften...                                    | ID_POPUP_PROJECT_PROJECTPROPERTIES      | 32428 |
| Entfernen                                           | ID_POPUP_PROJECT_REMOVE                 | 32308 |
| Auswahl hinzufügen/entfernen...                     | ID_POPUP_STRUCTURENODE_SELECTIONS       | 32491 |
|                                                     | ID_PREV_PANE                            | 32418 |
|                                                     | ID_TOGGLE_FOLDINGMARGIN                 | 32438 |
|                                                     | ID_TOGGLE_INDENTGUIDES                  | 32439 |
|                                                     | ID_TOGGLE_NUMLINEMARGIN                 | 32440 |
|                                                     | ID_WORD_WRAP                            | 32457 |

## 33.4 Aufrufen von MapForceAPI

Diese Dokumentation beschäftigt sich hauptsächlich mit den ActiveX Controls und Schnittstellen, die für die Integration der MapForce Benutzeroberfläche in Ihre Applikation benötigt werden. Über die folgenden Eigenschaften haben Sie Zugriff auf das MapForce Automation Interface (MapForceAPI), um die Funktionalitäten der integrierten Komponenten zu automatisieren oder zu definieren.

[MapForceControl.Application](#)

[MapForceControlDocument.Document](#)

[MapForceControlPlaceholder.Project](#)

Für die Verwendung des MapForce Automation Interface gelten beim Integrieren von MapForceControl auf Dokumentebene einige Einschränkungen. Nähere Informationen dazu finden Sie unter [Integration auf Dokumentebene](#).

## 33.5 Objektreferenz

Objekte:

[MapForceCommand](#)

[MapForceCommands](#)

[MapForceControl](#)

[MapForceControlDocument](#)

[MapForceControlPlaceholder](#)

Um Zugriff auf die MapForce Standardfunktionalitäten zu erhalten, können auch Objekte des **MapForce Automation Interface** aufgerufen werden. Nähere Informationen dazu finden Sie unter [MapForceControl.Application](#), [MapForceControlDocument.Document](#) und [MapForceControlPlaceholder.Project](#).

### 33.5.1 MapForceCommand

**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 folgende 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.

### 33.5.2 MapForceCommands

**Eigenschaften:**[Count](#)[Item](#)**Beschreibung:**

Sammlung von [Command](#)-Objekten für den Zugriff auf Befehlsbezeichnungen und IDs des MapForceControl. 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.

### 33.5.3 MapForceControl

**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 MapForce Bibliothek im Applikationsebenenmodus verwendet wird.

CLSID: A38637E9-5759-4456-A167-F01160CC22C1

ProgID: Altova.MapForceControl

**Eigenschaften**

Es sind die folgenden Eigenschaften definiert:

[IntegrationLevel](#)[EnableUserPrompts](#)[Appearance](#)[BorderStyle](#)

Befehlsbezogene Eigenschaften:

[CommandsList](#)[MainMenu](#)[Toolbars](#)

CommandsStructure (nicht mehr verwendet)

Zugriff auf die MapForceAPI:

[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 MapForce 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 MapForceControl verfügbar sind.

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 MapForceControl Objekts definiert werden.

MainMenu

**Eigenschaft:** MainMenu als [Command](#) (schreibgeschützt)

**Dispatch Id:** 1003

**Beschreibung:**

Mit dieser Eigenschaft erhalten Sie Zugriff auf die Beschreibung des MapForceControl Hauptmenüs.

Toolbars

**Eigenschaft:** Toolbars als [Commands](#) (schreibgeschützt)

**Dispatch Id:** 1005

**Beschreibung:**

Diese Eigenschaft gibt eine Liste aller Symbolleistenbeschreibungen zurück, die alle mit MapForceControl verfügbaren Symbolleisten beschreiben.

**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 MapForce 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 abruft, 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 Dokumentebene zu laden. Verwenden Sie statt dessen [MapForceControlDocument.Open](#) und [MapForceControlPlaceholder.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 MapForce Befehl erkannt wird. Wenn QueryStatus einen Wert 1 oder 5 zurückgibt, wird der Befehl deaktiviert.

**Events**

Das MapForceControl 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 MapForce 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 MapForceControl 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 in MapForce nicht verwendet**

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 MapForce 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 [MapForceControlDocument.OnDocumentOpened](#) zu verwenden.

OnFileChangedAlert

**Event:** OnFileChangedAlert (i\_strFilePath als [String](#)) als [bool](#)

**Dispatch Id: 1001****Beschreibung:**

Dieses Event wird ausgelöst, wenn eine mit MapForceControl geladene Datei auf der Festplatte von einer anderen Applikation geändert wurde. Clients sollten true zurückgeben, wenn Sie das Event behandelt haben oder false, wenn MapForce 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 MapForceControl 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 MapForce 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 MapForceControl 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 MapForce Befehls mittels [MapForceControl.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.

### 33.5.4 MapForceControlDocument

**Eigenschaften:**

[Appearance](#)  
[BorderStyle](#)  
[Document](#)  
[IsModified](#)  
[Path](#)  
[ReadOnly](#)

**Methoden:**

[Exec](#)  
[New](#)  
[Open](#)  
[QueryStatus](#)  
[Reload](#)  
[Save](#)  
[SaveAs](#)

**Events:**

[OnDocumentOpened](#)  
[OnDocumentClosed](#)  
[OnModifiedFlagChanged](#)  
[OnContextChanged](#)  
[OnFileChangedAlert](#)  
[OnActivate](#)

Wenn das MapForceControl im Modus auf Dokumentebene integriert ist, wird jedes Dokument in einem eigenen Objekt vom Typ MapForceControlDocument angezeigt. Das MapForceControlDocument enthält immer nur ein Dokument, kann aber wiederverwendet werden, um hintereinander mehrere Dateien anzuzeigen.

Dieses Objekt ist ein komplettes ActiveX Control.

CLSID: DFBB0871-DAFE-4502-BB66-08CEB7DF5255  
ProgID: Altova.MapForceControlDocument

**Eigenschaften**

Es sind die folgenden Eigenschaften definiert:

[ReadOnly](#)  
[IsModified](#)  
[Path](#)  
[Appearance](#)  
[BorderStyle](#)

Zugriff auf die MapForceAPI:  
[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 MapForce 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 MapForce 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 MapForce 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

**Methode:** `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 MapForceControlDocument 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:** Keine

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 MapForce 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 MapForce 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 MapForce 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.

### 33.5.5 MapForceControlPlaceholder

Eigenschaften für alle Arten von Platzhalterfenstern:

[PlaceholderWindowID](#)

Eigenschaften für das Projekt-Platzhalterfenster:

[Project](#)

Methoden für das Projekt-Platzhalterfenster:

[OpenProject](#)

[CloseProject](#)

Das MapForceControlPlaceholder Control dient dazu, die zusätzlichen MapForce 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: FDEC3B04-05F2-427d-988C-F03A85DE53C2  
ProgID: Altova.MapForceControlPlaceholder

#### Eigenschaften

Es sind die folgenden Eigenschaften definiert:

[PlaceholderWindowID](#)

Zugriff auf die MapForceAPI:

[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

**Dispatch Id:** 1

#### Beschreibung:

Mit Hilfe dieser Eigenschaft weiß das Objekt, welches MapForce Fenster im Client-Bereich des Control angezeigt werden soll. Die PlaceholderWindowID kann jederzeit auf jeden gültigen Wert der gesetzt werden. Das Control ändert seinen Status sofort und zeigt das neue MapForce Fenster an.

Project

**Eigenschaft:** Project als Project (schreibgeschützt)

**Dispatch Id: 2****Beschreibung:**

Mit der Eigenschaft `Project` erhalten Sie Zugriff auf das `Project` Objekt der MapForce 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 `MapForceXProjectWindow (=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 `MapForceXProjectWindow (=3)` übereinstimmt, schlägt die Methode fehl.

**Events**

Das MapForceControlPlaceholder 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 `MapForceXProjectWindow (=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.

### 33.5.6 Enumerationen

Es sind die folgenden Enumerationen definiert:

[ICActiveXIntegrationLevel](#)

[MapForceControlPlaceholderWindow](#)

#### **ICActiveXIntegrationLevel**

Mögliche Werte für die Eigenschaft [IntegrationLevel](#) für das MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel = 0
ICActiveXIntegrationOnDocumentLevel = 1
```

#### **MapForceControlPlaceholderWindow**

Diese Enumeration enthält die Liste der unterstützten zusätzlichen MapForce Fenster.

```
MapForceXNoWindow = -1
MapForceXLibraryWindow = 0
MapForceXOverviewWindow = 1
MapForceXValidationWindow = 2
MapForceXProjectWindow = 3
```





# Kapitel 34

---

## Anhänge

## 34 Anhänge

Diese Anhänge enthalten technische Informationen über MapForce und wichtige Lizenzinformationen. Die einzelnen Anhänge sind in die folgenden Unterabschnitte gegliedert:

### Technische Daten

- Betriebssystem und erforderlicher Arbeitsspeicher
- Altova XML Parser
- Altova XSLT- und XQuery-Prozessor
- Unicode-Unterstützung
- Internet-Verwendung
- Lizenzüberwachung

### Lizenzinformationen

- Electronic Software Distribution
- Copyright
- Endbenutzer-Lizenzvereinbarung (EULA)

## 34.1 Informationen zu den Prozessoren

Dieser Abschnitt enthält Informationen über implementierungsspezifische Funktionen des Altova XML Validators, des Altova XSLT 1.0-Prozessors, des Altova XSLT 2.0-Prozessors und des Altova XQuery-Prozessors.

### 34.1.1 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 `&#160;` (die Dezimalzeichenreferenz für ein geschütztes Leerzeichen) nicht als `&nbsp;`; 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 bold <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 bold <i> italic</i>.</para> or
<para>This is bold<#x20; <i>italic</i>.</para> or
<para>This is bold<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.

### 34.1.2 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](#).

#### 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-Konstrukturen 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:    | <a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>             |
| XPath 2.0-Funktionen | fn:    | <a href="http://www.w3.org/2005/xpath-functions">http://www.w3.org/2005/xpath-functions</a> |

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-Konstruktoren 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. Januar 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 bold <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 bold <i> italic</i>.</para> oder
<para>This is bold<#x20 <i>italic</i>.</para> oder
<para>This is bold<i><#x20italic</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.

### **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.



### 34.1.3 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:    | <a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>                           |
| Schema-Instanz      | xsi:   | <a href="http://www.w3.org/2001/XMLSchema-instance">http://www.w3.org/2001/XMLSchema-instance</a>         |
| Built-in Funktionen | fn:    | <a href="http://www.w3.org/2005/xpath-functions">http://www.w3.org/2005/xpath-functions</a>               |
| Lokale Funktionen   | local: | <a href="http://www.w3.org/2005/xquery-local-functions">http://www.w3.org/2005/xquery-local-functions</a> |

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 Beispielmódul:

```
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](#).

### 34.1.4 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.

#### 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 (zB. `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.

### **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      | <ul style="list-style-type: none"> <li>• 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.</li> <li>• Die Basis-URI eines Node im XML-Dokument kann mit Hilfe des Attributs <code>xml:base</code> geändert werden.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| collection    | <ul style="list-style-type: none"> <li>• Das Argument ist eine relative URI, die gegen die aktuelle Basis-URI aufgelöst wird.</li> <li>• 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> &lt;collection&gt;   &lt;doc href="uri-1" /&gt;   &lt;doc href="uri-2" /&gt;   &lt;doc href="uri-3" /&gt; &lt;/collection&gt; </pre> Die von den href Attributen referenzierten Dateien werden geladen und ihre Dokument-Nodes werden als Sequenz zurückgegeben. </li> <li>• Wenn mit der aufgelösten URI keine XML-Datei mit der oben beschriebenen Katalogstruktur identifiziert wird, dann wird der Argument-String (in dem die Platzhalterzeichen ? und * 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.</li> <li>• XSLT-Beispiel: Der Ausdruck <code>collection("c:\MyDocs\*.xml")//Title</code> gibt eine Sequenz aller DocTitle Elemente in den .xml Dateien im Ordner MyDocs zurück.</li> <li>• XQuery-Beispiel: Der Ausdruck <code>{for \$i in collection(c:\MyDocs\*.xml) return element doc{base-uri(\$i)}}</code> gibt die Basis-URIs aller .xml Dateien im Ordner MyDocs zurück, wobei jede URI sich innerhalb eines doc Elemente befindet.</li> <li>• Die Standard-Collection ist leer.</li> </ul> |
| count         | <ul style="list-style-type: none"> <li>• Siehe Anmerkungen zu Whitespaces im Abschnitt <a href="#">Allgemeine Informationen</a>.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

|                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| current-date,<br>current-dateTime,<br>current-time | <ul style="list-style-type: none"> <li>• The argument is a relative URI that is resolved against the current base URI.</li> <li>• 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> &lt;collection&gt;   &lt;doc href="uri-1" /&gt;   &lt;doc href="uri-2" /&gt;   &lt;doc href="uri-3" /&gt; &lt;/collection&gt; </pre> The files referenced by the href attributes are loaded, and their document nodes are returned as a sequence. </li> <li>• 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.</li> <li>• The default collection is empty.</li> </ul> |
| deep-equal                                         | <ul style="list-style-type: none"> <li>• Siehe Anmerkung zu Whitespace im Abschnitt <a href="#">Allgemeine Informationen</a>.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| doc                                                | <ul style="list-style-type: none"> <li>• 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.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| id                                                 | <ul style="list-style-type: none"> <li>• In einem wohlgeformten aber ungültigen Dokument, das zwei oder mehr Elemente mit demselben ID-Wert enthält, wird das erste Element in der Dokumentenreihenfolge zurückgegeben.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| in-scope-prefixes                                  | <ul style="list-style-type: none"> <li>• 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.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| last                                               | <ul style="list-style-type: none"> <li>• Siehe Anmerkung zu Whitespace im Abschnitt <a href="#">Allgemeine Informationen</a>.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| lower-case                                         | <ul style="list-style-type: none"> <li>• Es wird der Unicode-Zeichensatz unterstützt.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| normalize-unicode                                  | <ul style="list-style-type: none"> <li>• Es werden die Normalisierungsformulare NFC, NFD, NFKC und NFKD unterstützt.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| position                                           | <ul style="list-style-type: none"> <li>• Siehe Anmerkung zu Whitespace im Abschnitt <a href="#">Allgemeine Informationen</a>.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| resolve-uri     | <ul style="list-style-type: none"><li>• 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.</li><li>• Die relative URI (das erste Argument) wird nach dem letzten "/" in der Pfadnotation der Basis-URI-Notation angehängt.</li><li>• 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).</li></ul> |
| static-base-uri | <ul style="list-style-type: none"><li>• Die Basis-URI aus dem statischen Kontext ist die Basis-URI des XSLT Stylesheets oder die im Prolog des XQuery-Dokuments angegebene Basis-URI.</li><li>• Bei Verwendung des XPath Evaluators in der XMLSpy IDE ist die Basis-URI aus dem statischen Kontext die URI des aktiven XML-Dokuments.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                               |
| upper-case      | <ul style="list-style-type: none"><li>• Der Unicode-Zeichensatz wird unterstützt.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |



### 34.1.5 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.

#### 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: Constructoren](#)
- [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 Methodenaufruf als Präfix vorangestellt wird.

#### Klassendatei in einem Paket, XSLT/XQuery-Datei befindet sich im selben Ordner wie das Java-Paket

Im Beispiel unten wird die Methode `getVehicleType()` der Klasse `car` des Pakets `com.altova.extfunc` aufgerufen. Das Paket `com.altova.extfunc` befindet sich im Ordner `JavaProject`. Die XSLT-Datei befindet sich ebenfalls im Ordner `JavaProject`.

```
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:fn="http://www.w3.org/2005/xpath-functions"
 xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
 <a>
 <xsl:value-of select="car:getVehicleType()"/>

</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()"/>

</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)"/>
</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 Methodenaufruf als Präfix vorangestellt wird. Im Beispiel unten sehen Sie, wie eine Klassendatei aufgerufen wird, die sich in einem anderen als dem aktuellen Verzeichnis befindet.

```
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:fn="http://www.w3.org/2005/xpath-functions"
 xmlns:car="
java:Car?path=file:///C:/JavaProject/com/altova/extfunc/" >

 <xsl:output exclude-result-prefixes="fn car xsl xs"/>

 <xsl:template match="/">
 <xsl:variable name="myCar" select="car:new('red')"/>
 <a><xsl:value-of select="car:getCarColor($myCar)"/>
 </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 Methodenaufruf angegeben werden. Hier sehen Sie zwei Beispiele für die Syntax:

```
xmlns:ns1="java:docx.layout.pages?path=jar:file:///c:/projects/docs/docx.jar!/"
"
 ns1:main()

xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
```

```
ns2:docx.layout.pages.main()
```

Hier sehen Sie ein komplettes XSLT-Beispiel, in dem eine JAR-Datei verwendet wird, um eine Java-Erweiterungsfunktion aufzurufen.:

```
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:fn="http://www.w3.org/2005/xpath-functions"
 xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
 <xsl:variable name="myCar" select="car:Car1.new('red')"/>
 <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/>
</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. `(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:jl="java:java.lang">
 <xsl:param name="CurrentDate" select="date:new()" />
 <xsl:template match="/">
 <enrollment institution-id="Altova School"
 date="{date:toString($CurrentDate)}"
 type="{jl:lang:Object.toString(jl:lang: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 `jl: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</code> (primitive), <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</code> (primitive), <code>java.lang.Float</code> , <code>double</code> (primitive)                                                                                                 |
| <code>xs:double</code>  | <code>double</code> (primitive), <code>java.lang.Double</code>                                                                                                                                 |
| <code>xs:decimal</code> | <code>float</code> (primitive), <code>java.lang.Float</code> , <code>double</code> (primitive), <code>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.

#### .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, 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. Diese Objekt wird zweimal erstellt, einmal als Wert der Variablen `releasedate`, ein zweites Mal als das erste und einzige Argument der Methode `System.DateTime.ToString()`. Die Instanzmethode `System.DateTime.ToString()` wird zwei Mal aufgerufen, beide Male mit dem `System.DateTime` Konstruktor (`new(2008, 4, 29)`) als erstem und einzigem Argument. In einer dieser Instanzen wird die Variable `releasedate` verwendet, um das .NET-Objekt abzurufen.

### Instanzmethoden und Instanzfelder

Der Unterschied zwischen einer Instanzmethode und einem Instanzfeld ist ein theoretischer. In einer Instanzmethode wird ein .NET-Objekt direkt als Argument übergeben; in einem

Instanzfeld wird stattdessen ein Parameter oder eine Variable übergeben - auch wenn der Parameter bzw. die Variable selbst ein .NET-Objekt enthalten kann. So enthält z.B. die Variable `releasedate` im Beispiel oben ein .NET-Objekt und es ist diese Variable, die als das Argument von `ToString()` an den zweiten `date` Elementkonstruktor übergeben wird. Die `ToString()` Instanz im ersten `date` Element ist daher eine Instanzmethode, während die zweite als Instanzfeld betrachtet wird. Das in beiden Instanzen erzeugte Ergebnis ist jedoch dasselbe.

#### 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.

### 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>
 Total Retail Price =
 $<xsl:value-of select="user:AddMargin(50)" />

 Total Wholesale Price =
 $<xsl:value-of select="50" />

 </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.

### 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.

**Diagrammfunktionen**

Mit Hilfe der unten aufgelisteten Diagrammfunktionen können Sie Diagramme als Bilder erstellen, generieren und speichern. Sie werden in der aktuellen Version Ihres Altova-Produkts auf die unten beschriebene Art unterstützt. Beachten Sie jedoch, dass eine oder mehrere dieser Funktionen in zukünftigen Produktversionen eventuell nicht mehr unterstützt werden, bzw. dass sich das Verhalten einzelner Funktionen ändern kann. Um Informationen über die Unterstützung für Altova Erweiterungsfunktionen in der jeweiligen Release zu erhalten, schlagen Sie bitte in der Dokumentation der jeweils aktuellen Release nach.

Die Diagrammfunktionen werden in zwei Gruppen unterteilt:

- [Funktionen zum Generieren und Speichern von Diagrammen](#)
- [Funktionen zur Erstellung von Diagrammen](#)

**Anmerkung:** Diagrammfunktionen werden nur in der **Enterprise und der Reporting Edition** von Altova-Produkten unterstützt.

**Funktionen zum Generieren und Speichern von Diagrammen**

Diese Funktionen generieren anhand des (mit Hilfe der Diagrammerstellungsfunktionen) erzeugten Diagrammobjekts entweder ein Bild oder speichern ein Bild in einer Datei

**altova:generate-chart-image** (\$chart, \$width, \$height, \$encoding) als atomic

wobei

- \$chart ist das Diagrammerweiterungsobjekt, das Sie mit der Funktion altova:create-chart erhalten
- \$width und \$height muss mit einer Längeneinheit definiert werden
- \$encoding kann base64Binary oder binarytobase16 sein

Die Funktion gibt das Diagrammbild in der definierten Kodierung zurück.

**altova:generate-chart-image** (\$chart, \$width, \$height, \$encoding, \$imagetype)  
als atomic

wobei

- \$chart das Diagramm-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart erzeugt wurde
- \$width und \$height mit einer Längeneinheit definiert werden müssen
- \$encoding den Wert base64Binary oder hexBinary haben kann

- `$imagetype` eines der folgenden Bildformate sein kann: png, gif, bmp, jpg, jpeg

Die Funktion gibt das Diagrammbild in der definierten Kodierung und im definierten Bildformat zurück.

**altova:save-chart-image** (`$chart`, `$filename`, `$width`, `$height`) als `empty()`

wobei

- `$chart` das Diagramm-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart` erzeugt wurde
- `$filename` der Pfad und Name der Datei ist, unter dem das Diagrammbild gespeichert werden soll
- `$width` und `$height` mit einer Längeneinheit definiert werden müssen

Die Funktion speichert das Diagrammbild unter dem in `$filename` definierten Dateinamen.

**altova:save-chart-image** (`$chart`, `$filename`, `$width`, `$height`, `$imagetype`) als `empty()`

wobei

- `$chart` das Diagramm-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart` erzeugt wurde
- `$filename` der Pfad und Name der Datei ist, unter dem das Diagrammbild gespeichert werden soll
- `$width` und `$height` mit einer Längeneinheit definiert werden müssen
- `$imagetype` eines der folgenden Bildformate sein kann: png, gif, bmp, jpg, jpeg

Die Funktion speichert das Diagrammbild im definierten Bildformat unter dem in `$filename` definierten Dateinamen.

### Funktionen zur Erstellung von Diagrammen

Die folgenden Funktionen dienen zur Erstellung von Diagrammen.

**altova:create-chart** (`$chart-config`, `$chart-data-series*`) als `chart extension item`

wobei

- `$chart-config` das Diagrammkonfigurations-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart-config` oder über die Funktion `altova:create-chart-config-from-xml` erzeugt wurde
- `$chart-data-series` ist das `chart-data-series` Erweiterungsobjekt, das mit der Funktion `altova:create-chart-data-series` oder mit der Funktion `altova:create-chart-config-from-rows` erzeugt wurde

Die Funktion gibt ein Diagrammerweiterungsobjekt zurück, das anhand der über die Argumente gelieferten Daten erzeugt wird.

**altova:create-chart-config** (`$type-name`, `$title`) als `chart-config Erweiterungsobjekt`

wobei

- `$type-name` den Typ des zu erstellenden Diagramms definiert: `Pie`, `Pie3d`, `BarChart`, `BarChart3d`, `BarChart3dGrouped`, `LineChart`, `ValueLineChart`, `RoundGauge`, `BarGauge`
- `$title` der Name des Diagramms ist

Die Funktion gibt ein Diagrammkonfigurations-Erweiterungsobjekt zurück, das die Konfigurationsinformationen zum Diagramm enthält.

**altova:create-chart-config-from-xml**(`$xml-struct`) als `chart-config` Erweiterungsjekt

wobei

- `$xml-struct` die XML-Struktur ist, die die Konfigurationsinformationen des Diagramms enthält

Die Funktion gibt ein Diagrammkonfigurations-Erweiterungsobjekt zurück, das die Konfigurationsinformationen zum Diagramm enthält. Diese Informationen werden in einem [XML-Datenfragment](#) geliefert.

**altova:create-chart-data-series**(`$series-name?`, `$x-values*`, `$y-values*`) als `chart-data-series` Erweiterungsjekt

wobei

- `$series-name` der Name der Datenreihe ist
- `$x-values` die Liste der Werte für die X-Achse liefert
- `$y-values` die Liste der Werte für die Y-Achse liefert

Die Funktion gibt ein Diagrammdatenreihen-Erweiterungsobjekt zurück, das die Daten zur Erstellung des Diagramms, also die Namen der Datenreihen, und die Achsendaten enthält.

**altova:create-chart-data-row**(`x`, `y1`, `y2`, `y3`, ...) als `chart-data-x-Ny-row` Erweiterungsjekt

wobei

- `x` der Wert der X-Achsen-Spalte der Diagrammdatenzeile ist
- `yN` die Werte der Spalten für die Y-Achse sind

Die Funktion gibt ein `chart-data-x-Ny-row` Erweiterungsjekt zurück, das die Daten für die X-Achsen-Spalte und die Y-Achsen-Spalten einer einzigen Datenreihe enthält.

**altova:create-chart-data-series-from-rows**(`$series-names` als `xs:string*`, `$row*`) als `chart-data-series` Erweiterungsjekt

wobei

- `$series-name` der Name der zu erstellenden Datenreihen ist
- `$row` das `chart-data-x-Ny-row` Erweiterungsjekt ist, das als Datenreihe erstellt werden soll

Die Funktion gibt ein `chart-data-series` Erweiterungsjekt zurück, das die Daten für die X-

und die Y-Achse der Datenreihe enthält.

**altova:create-chart-layer**(\$chart-config, \$chart-data-series\*) als chart-layer Erweiterungsobjekt

wobei

- \$chart-config das chart-config-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml abgerufen wird
- \$chart-data-series das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-data-series oder der Funktion altova:create-chart-data-series-from-rows abgerufen wird

Die Funktion gibt ein chart-layer Erweiterungsobjekt zurück, das chart-layer-Daten enthält.

**altova:create-multi-layer-chart**(\$chart-config, \$chart-data-series\*, \$chart-layer\*)

wobei

- \$chart-config das chart-config Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml abgerufen wird
- \$chart-data-series das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-data-series oder der Funktion altova:create-chart-data-series-from-rows abgerufen wird
- \$chart-layer das chart-layer-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-layer abgerufen wird

Die Funktion gibt ein multi-layer-chart-Objekt zurück.

**altova:create-multi-layer-chart**(\$chart-config, \$chart-data-series\*, \$chart-layer\*, xs:boolean \$mergecategoryvalues)

wobei

- \$chart-config das chart-config Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml abgerufen wird
- \$chart-data-series das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-data-series oder der Funktion altova:create-chart-data-series-from-rows abgerufen wird
- \$chart-layer das chart-layer-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-layer abgerufen wird

Die Funktion gibt ein multi-layer-chart-Objekt zurück.

#### Diagrammdaten-XML-Struktur

Unten sehen Sie die XML-Struktur von Diagrammdaten, wie sie für [Altova-Erweiterungsfunktionen für Diagramme](#) angezeigt werden könnte. Diese Funktionen beeinflussen das Aussehen der einzelnen Diagramme. Nicht alle Elemente werden für alle Diagrammart verwendet, so wird z.B. das Element <Pie> bei Balkendiagrammen ignoriert.



**Anmerkung:** Diagrammfunktionen werden nur in der **Enterprise und der Reporting Edition** von Altova Produkten unterstützt.

#### <chart-config>

##### <General

```

 SettingsVersion="1" must be provided
 ChartKind="BarChart" Pie, Pie3d, BarChart, StackedBarChart, BarChart3d,
BarChart3dGrouped, LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge,
BarGauge, CandleStick
 BKColor="#ffffff" Color
 BKColorGradientEnd="#ffffff" Color. In case of a gradient, BKColor and
BKColorGradientEnd define the gradient's colors
 BKMode="#ffffff" Solid, HorzGradient, VertGradient
 BKFile="Path+Filename" String. If file exists, its content is drawn over the
background.
 BKFileMode="Stretch" Stretch, ZoomToFit, Center, Tile
 ShowBorder="1" Bool
 PlotBorderColor="#000000" Color
 PlotBKColor="#ffffff" Color
 Title="" String
 ShowLegend="1" Bool
 OutsideMargin="3.%" PercentOrPixel
 TitleToPlotMargin="3.%" PercentOrPixel
 LegendToPlotMargin="3.%" PercentOrPixel
 Orientation="vert" Enumeration: possible values are: vert, horz
>
<TitleFont
 Color="#000000" Color
 Name="Tahoma" String
 Bold="1" Bool
 Italic="0" Bool
 Underline="0" Bool
 MinFontHeight="10.pt" FontSize (only pt values)
 Size="8.%" FontSize />
<LegendFont
 Color="#000000"
 Name="Tahoma"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="3.5%" />
<AxisLabelFont
 Color="#000000"
 Name="Tahoma"
 Bold="1"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="5.%" />
</General>

<Line
 ConnectionShapeSize="1.%" PercentOrPixel
 DrawFilledConnectionShapes="1" Bool
 DrawOutlineConnectionShapes="0" Bool
 DrawSlashConnectionShapes="0" Bool

```

```

 DrawBackslashConnectionShapes="0" Bool
 />

 <Bar
 ShowShadow="1" Bool
 ShadowColor="#a0a0a0" Color
 OutlineColor="#000000" Color
 ShowOutline="1" Bool
 />

 <Area
 Transparency="0" UINT (0-255) 255 is fully transparent, 0 is opaque
 OutlineColor="#000000" Color
 ShowOutline="1" Bool
 />

 <CandleStick
 FillHighClose="0" Bool. If 0, the body is left empty. If 1, FillColorHighClose is used
 for the candle body
 FillColorHighClose="#ffffff" Color. For the candle body when close > open
 FillHighOpenWithSeriesColor="1" Bool. If true, the series color is used to fill the
 candlebody when open > close
 FillColorHighOpen="#000000" Color. For the candle body when open > close and
 FillHighOpenWithSeriesColor is false
 />

 <Colors User-defined color scheme: By default this element is empty except for the style
 and has no Color attributes
 UseSubsequentColors ="1" Boolean. If 0, then color in overlay is used. If 1, then
 subsequent colors from previous chart layer is used
 Style="User" Possible values are: "Default", "Grayscale", "Colorful", "Pastel", "User"
 Colors="#52aca0" Color: only added for user defined color set
 Colors1="#d3c15d" Color: only added for user defined color set
 Colors2="#8971d8" Color: only added for user defined color set
 ...
 ColorsN="" Up to ten colors are allowed in a set: from Colors to Colors9
 </Colors>

 <Pie
 ShowLabels="1" Bool
 OutlineColor="#404040" Color
 ShowOutline="1" Bool
 StartAngle="0." Double
 Clockwise="1" Bool
 Draw2dHighlights="1" Bool
 Transparency="0" Int (0 to 255: 0 is opaque, 255 is fully transparent)
 DropShadowColor="#c0c0c0" Color
 DropShadowSize="5.%" PercentOrPixel
 PieHeight="10.%" PercentOrPixel. Pixel values might be different in the result
 because of 3d tilting
 Tilt="40.0" Double (10 to 90: The 3d tilt in degrees of a 3d pie)
 ShowDropShadow="1" Bool
 ChartToLabelMargin="10.%" PercentOrPixel
 AddValueToLabel="0" Bool
 AddPercentToLabel="0" Bool
 AddPercentToLabels_DecimalDigits="0" UINT (0 – 2)
 >
 <LabelFont

```

```

 Color="#000000"
 Name="Arial"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="4.%" />
 </Pie>

 <XY>
 <XAxis Axis
 AutoRange="1" Bool
 AutoRangeIncludesZero="1" Bool
 RangeFrom="0." Double: manual range
 RangeTill="1." Double: manual range
 LabelToAxisMargin="3.%" PercentOrPixel
 AxisLabel="" String
 AxisColor="#000000" Color
 AxisGridColor="#e6e6e6" Color
 ShowGrid="1" Bool
 UseAutoTick="1" Bool
 ManualTickInterval="1." Double
 AxisToChartMargin="0.px" PercentOrPixel
 TickSize="3.px" PercentOrPixel
 ShowTicks="1" Bool
 ShowValues="1" Bool
 AxisPosition="LeftOrBottom" Enums: "LeftOrBottom", "RightOrTop",
"AtValue"
 AxisPositionAtValue = "0" Double
 >
 <ValueFont
 Color="#000000"
 Name="Tahoma"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="3.%" />
 </XAxis>
 <YAxis Axis (same as for XAxis)
 AutoRange="1"
 AutoRangeIncludesZero="1"
 RangeFrom="0."
 RangeTill="1."
 LabelToAxisMargin="3.%"
 AxisLabel=""
 AxisColor="#000000"
 AxisGridColor="#e6e6e6"
 ShowGrid="1"
 UseAutoTick="1"
 ManualTickInterval="1."
 AxisToChartMargin="0.px"
 TickSize="3.px"
 ShowTicks="1" Bool
 ShowValues="1" Bool
 AxisPosition="LeftOrBottom" Enums: "LeftOrBottom", "RightOrTop",
"AtValue"
 AxisPositionAtValue = "0" Double
 >
 <ValueFont
 Color="#000000"

```

```

 Name="Tahoma"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="3.%"/>
 </YAxis>
</XY>

<XY3d
 AxisAutoSize="1" Bool: If false, XSize and YSize define the aspect ration of x and y
axis. If true, aspect ratio is equal to chart window
 XSize="100.%" PercentOrPixel. Pixel values might be different in the result because
of 3d tilting and zooming to fit chart
 YSize="100.%" PercentOrPixel. Pixel values might be different in the result because
of 3d tilting and zooming to fit chart
 SeriesMargin="30.%" PercentOrPixel. Pixel values might be different in the result
because of 3d tilting and zooming to fit chart
 Tilt="20." Double. -90 to +90 degrees
 Rot="20." Double. -359 to +359 degrees
 FoV="50."> Double. Field of view: 1-120 degree
>
 <ZAxis
 AutoRange="1"
 AutoRangeIncludesZero="1"
 RangeFrom="0."
 RangeTill="1."
 LabelToAxisMargin="3.%"
 AxisLabel=""
 AxisColor="#000000"
 AxisGridColor="#e6e6e6"
 ShowGrid="1"
 UseAutoTick="1"
 ManualTickInterval="1."
 AxisToChartMargin="0.px"
 TickSize="3.px" >
 <ValueFont
 Color="#000000"
 Name="Tahoma"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="3.%"/>
 </ZAxis>
 </XY3d>

<Gauge
 MinVal="0." Double
 MaxVal="100." Double
 MinAngle="225" UINT: -359-359
 SweepAngle="270" UINT: 1-359
 BorderToTick="1.%" PercentOrPixel
 MajorTickWidth="3.px" PercentOrPixel
 MajorTickLength="4.%" PercentOrPixel
 MinorTickWidth="1.px" PercentOrPixel
 MinorTickLength="3.%" PercentOrPixel
 BorderColor="#a0a0a0" Color
 FillColor="#303535" Color
 MajorTickColor="#a0c0b0" Color
 MinorTickColor="#a0c0b0" Color

```

```

 BorderWidth="2.%" PercentOrPixel
 NeedleBaseWidth="1.5%" PercentOrPixel
 NeedleBaseRadius="5.%" PercentOrPixel
 NeedleColor="#f00000" Color
 NeedleBaseColor="#141414" Color
 TickToTickValueMargin="5.%" PercentOrPixel
 MajorTickStep="10." Double
 MinorTickStep="5." Double
 RoundGaugeBorderToColorRange="0.%" PercentOrPixel
 RoundGaugeColorRangeWidth="6.%" PercentOrPixel
 BarGaugeRadius="5.%" PercentOrPixel
 BarGaugeMaxHeight="20.%" PercentOrPixel
 RoundGaugeNeedleLength="45.%" PercentOrPixel
 BarGaugeNeedleLength="3.%" PercentOrPixel
 >
 <TicksFont
 Color="#a0c0b0"
 Name="Tahoma"
 Bold="0"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="4.%"
 />
 <ColorRanges> User-defined color ranges. By default empty with no child element
entries
 <Entry
 From="50." Double
 FillWithColor="1" Bool
 Color="#00ff00" Color
 />
 <Entry
 From="50.0"
 FillWithColor="1"
 Color="#ff0000"
 />
 ...
 </ColorRanges>
</Gauge>
</chart-config>

```

Beispiel: Diagrammfunktionen

Anhand des XSLT-Beispieldokuments weiter unten sehen Sie, wie [Altova-Erweiterungsfunktionen für Diagramme](#) eingesetzt werden können. Weiter unten sehen Sie ein XML-Dokument und eine Abbildung des Ausgabebilds, das generiert wird, wenn ein XML-Dokument mit dem Altova XSLT 2.0-Prozessor anhand des XSLT-Dokuments verarbeitet wird.

**Anmerkung:** Diagrammfunktionen werden nur in der **Enterprise und der Reporting Edition** von Altova Produkten unterstützt.

**Anmerkung:** Weitere Informationen zur Erstellung von Diagrammdatentabellen finden Sie in der Dokumentation zu den Altova-Produkten [XMLSpy](#) und [StyleVision](#).

### XSLT-Dokument

In diesem (*unten aufgelisteten*) XSLT-Dokument wird mit Hilfe der Altova Diagramm-Erweiterungsfunktionen ein Kreisdiagramm generiert. Das XSLT-Dokument kann zur

Verarbeitung des weiter unten angeführten XML-Dokuments verwendet werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:altovaext="http://www.altova.com/xslt-extensions"
 exclude-result-prefixes="#all">
 <xsl:output version="4.0" method="html" indent="yes" encoding="UTF-8"/>
 <xsl:template match="/">
 <html>
 <head>
 <title>
 <xsl:text>HTML Page with Embedded Chart</xsl:text>
 </title>
 </head>
 <body>
 <xsl:for-each select="/Data/Region[1]">
 <xsl:variable name="extChartConfig" as="item()*">
 <xsl:variable name="ext-chart-settings" as="item()*">
 <chart-config>
 <General
 SettingsVersion="1"
 ChartKind="Pie3d"
 BKColor="#ffffff"
 ShowBorder="1"
 PlotBorderColor="#000000"
 PlotBKColor="#ffffff"
 Title="{@id}"
 ShowLegend="1"
 OutsideMargin="3.2%"
 TitleToPlotMargin="3.%"
 LegendToPlotMargin="6.%"
 >
 <TitleFont
 Color="#023d7d"
 Name="Tahoma"
 Bold="1"
 Italic="0"
 Underline="0"
 MinFontHeight="10.pt"
 Size="8.%" />
 </General>
 </chart-config>
 </xsl:variable>
 <xsl:sequence select="
altovaext:create-chart-config-from-xml($ext-chart-settings)"/>
 </xsl:variable>
 <xsl:variable name="chartDataSeries" as="item()*">
 <xsl:variable name="chartDataRows" as="item()*">
 <xsl:for-each select="(Year)">
 <xsl:sequence select="
altovaext:create-chart-data-row({@id}, (.))"/>
 </xsl:for-each>
 </xsl:variable>
 <xsl:variable name="chartDataSeriesNames" as="xs:string*"
select=" (("Series 1"), '') [1]"/>
 <xsl:sequence
 select="altovaext:create-chart-data-series-from-rows(
$chartDataSeriesNames, $chartDataRows)"/>
 </xsl:variable>
 <xsl:variable name="ChartObj" select="altovaext:create-chart(
$extChartConfig, ($chartDataSeries), false())"/>
 <xsl:variable name="sChartFileName" select="'mychart1.png'"/>

 </xsl:for-each>
 </body>
 </html>
 </template>
</xsl:stylesheet>
```

```

 </body>
 </html>
</xsl:template>
</xsl:stylesheet>

```

### XML-Dokument

Dieses XML-Dokument kann mit dem oben stehenden XSLT-Dokument verarbeitet werden. Anhand der Daten im XML-Dokument wird das in der unten stehenden Abbildung gezeigte Kreisdiagramm generiert.

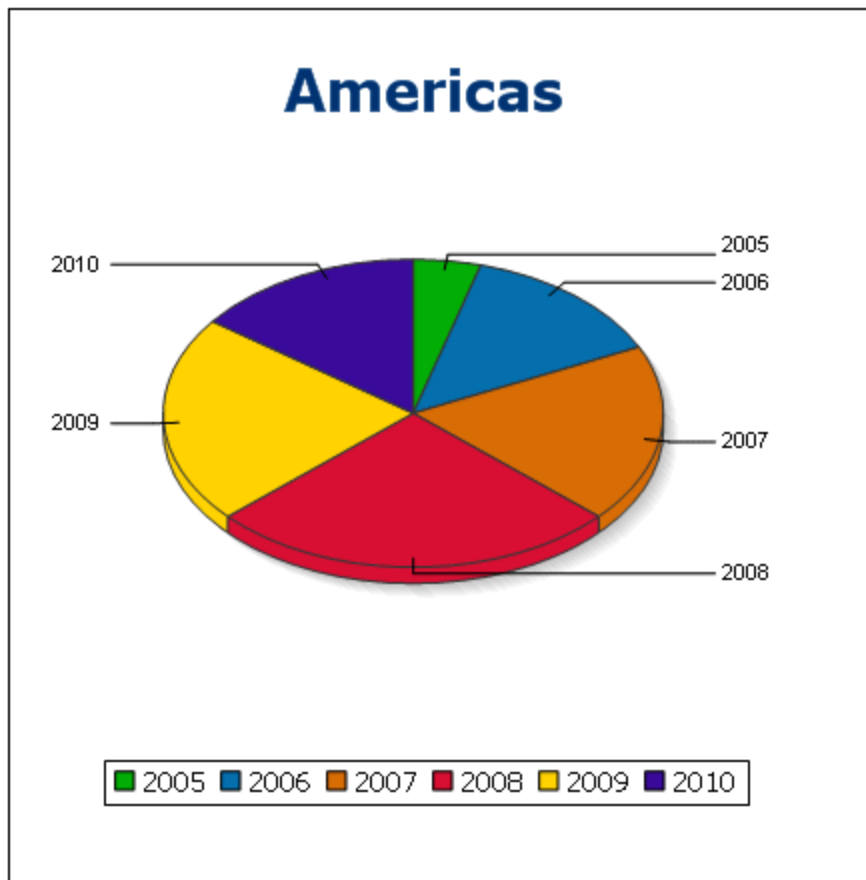
```

<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="YearlySales.xsd">
 <ChartType>Pie Chart 2D</ChartType>
 <Region id="Americas">
 <Year id="2005">30000</Year>
 <Year id="2006">90000</Year>
 <Year id="2007">120000</Year>
 <Year id="2008">180000</Year>
 <Year id="2009">140000</Year>
 <Year id="2010">100000</Year>
 </Region>
 <Region id="Europe">
 <Year id="2005">50000</Year>
 <Year id="2006">60000</Year>
 <Year id="2007">80000</Year>
 <Year id="2008">100000</Year>
 <Year id="2009">95000</Year>
 <Year id="2010">80000</Year>
 </Region>
 <Region id="Asia">
 <Year id="2005">10000</Year>
 <Year id="2006">25000</Year>
 <Year id="2007">70000</Year>
 <Year id="2008">110000</Year>
 <Year id="2009">125000</Year>
 <Year id="2010">150000</Year>
 </Region>
</Data>

```

### Ausgabebild

Das unten gezeigte Kreisdiagramm wird generiert, wenn das oben aufgelistete XML-Dokument mit Hilfe des XSLT-Dokuments verarbeitet wird.





## 34.2 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](#)

### 34.2.1 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.

### 34.2.2 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.

### 34.2.3 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.

### 34.2.4 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.

#### **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.

### **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.

### 34.2.5 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..

### 34.3 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.



### 34.3.1 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](http://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.

### 34.3.2 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.

### 34.3.3 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](http://www.altova.com/legal_3rdparty.html) aufgelistet.

Alle anderen Namen oder Warenzeichen sind das Eigentum ihrer jeweiligen Inhaber.

### 34.3.4 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](http://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 Vertriebs Händler 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 dekompile, 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 HAFTUNGSSAUSCHLUSS, 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 Projektdaten 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 DURCHGEFÜHRTE 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](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

(

**(Standard),**

mehrere Input- / Output-Komponenten, 55

▪

**.NET,**

Unterschiede zu XMLSPY Standalone, 881

**.NET 2002/2003, 975**

**.NET Erweiterungsfunktionen,**

Datentypkonvertierungen, .NET in XPath/XQuery, 1273

Datentypkonvertierungen, XPath/XQuery in .NET, 1272  
für XSLT und XQuery, 1267

Instanzmethoden, Instanzfelder, 1271

Konstruktoren, 1269

statische Methoden, statische Felder, 1270

Übersicht, 1267

<

**<dynamic>,**

Datei-Input / Output, 55

## 0

**0D und 0A,**

Sonderzeichen ersetzen, 253

## 2

**2005R3,**

Kompatibilitätsmodus, 1032

**2012, 1063**

## 8

**8601,**

ISO Datum - Uhrzeit, 607

## 9

**997,**

X12 Functional Acknowledgement, 598

X12 Funktionsbestätigung, 598

**999,**

X12 Implementation Acknowledgement, 600

## A

**A bis Z,**

Sortierkomponente, 236

**aar-Datei,**

Axis-Archiv für Axis2, 825

**Abfrage,**

Datenbankabfrage, 460

XML-Daten in DB2, 448

**Abruf,**

Modus, 469

**abs, 794, 808**

**Absolut,**

Pfade - Vorteile/Nachteile, 205

**Access,**

auf IBM DB2 mappen, 452

MS Access-Unterstützung, 338

**Acknowledgement,**

X12 997, 598

X12 999, 600

**Acknowledgment,**

X12 997/999 - TA1 Segment, 602

**acos, 794**

**Active, 1131**

**ActiveDocument, 1090, 1125**

**add, 770**

**Adjust-to-Timezone, 805**

**Aggregat,**

Funktionen, 255

**Aggregieren,**

Funktion, 730

**Aktion,**

Ignorieren - Tabellenaktion, 383

Symbol "Tabellenaktionen", 367

Tabellenaktion und Transaktionen, 400

Tabellenaktionen, 398

Tabellendaten vor Aktion löschen, 398

**Aktualisieren,**

Datenbank, 469

Datenbankaktion, 367

und Child löschen, 375

**Alle Concepts,**

XBRL-Komponenteneinstellungen, 851

**Alle Concepts (nicht hierarchisch),**

XBRL-Komponenteneinstellungen, 851

**Alle einfügen,**

keine Tabellenaktionen nach "Alle einfügen", 400

Tabellenaktion - Spalten rechts löschen, 400

**Alles kopieren,**

Konnektoren auflösen / löschen, 109

und Filter, 109

Verbindungen, 109

**Allgemeine,**

Optionen Datenbankabfrage, 486

**Als CSV speichern, 570****Als FLF speichern, 576****Altova Erweiterungen,**

Diagrammfunktionen (siehe Diagrammfunktionen), 1275

**Altova Erweiterungsfunktionen,**

allgemeine Funktionen, 1276

Diagrammfunktionen (siehe Diagrammfunktionen), 1276

**Altova Website, 963****Altova XML Parser,**

Info, 1293

**Altova XSLT 1.0-Prozessor,**

Einschränkungen und implementierungsspezifisches Verhalten, 1246

**Altova XSLT 2.0-Prozessor,**

allgemeine Informationen, 1248

Informationen, 1248

**Altova-Prozessoren,**

in Altova-Produkten, 1294

**Ändern,**

Datenbankkomponente, 350

**Anführungszeichen,**

und Befehlszeilenparameter, 273

**Annehmen,**

Trennzeichen vorhanden, 522

**Anpassen,**

ANSI X12-Transaktion, 627

EDIFACT, 617

**Anpassung,**

EDIFACT global, 621

EDIFACT inline, 623

X12 global, 632

X12 Vorbereitung, 630

**ANSI X12,**

als Ziel, 607

Autokomplettierungsregeln, 614

Vallidierungsregeln - Ziel, 614

**Ansicht, 948**

von MapForce, 1131

**ANT,**

Java-Webservice kompilieren, 825

Script, 973

**Anweisung,**

SQL ausführen, 472

**anyURI,**

Funktionen, 802

**API,**

aufrufen, 1221

Dokumentation, 1058

über die ODBC API verbinden, 387

Übersicht, 1059

**Application-Objekt, 1090****Applikation, 1089**

für Dokumente, 1125

**Applikationsarbeitsablauf,**

bei Verwendung globaler Ressourcen, 171

**Applikationsebene,**

Integration von MapForce, 1185

**Applikationsname, 1032****Applikationsobjekt, 1060****AppOutputLine, 1097****AppOutputLines, 1102****AppOutputLineSymbol, 1104****Arabisch, 1296****Arbeitsablauf,**

bei Verwendung globaler Ressourcen, 171

starten - globale Ressource, 175

**Arbeitsblatt,**

Excel 2007 - Datenelemente definieren, 659

in XML-Output-Dateien, 200

**Arbeitsmappe,**

Excel 2007 - Datenelemente definieren, 659

**Arbeitsspeicher,**

Ausnahmeereignis bei zu wenig, 1030

**Arbeitsspeicher-Anforderungen, 1292****Archiv,**

Axis2 aar-Datei, 825

**Arial Unicode MS, 1295****asin, 794****atan, 794****Atomisierung von Nodes,**

in XPath 2.0- und XQuery 1.0-Auswertung, 1254

**ATTLIST,**

DTD Namespace URIs, 294

**Aufruf,**

Webservice Fault, 837

WSDL-Aufruf-Einstellungen, 837

**Aufrufen,**

Template, 724

**Aufrufen von Webservices, 836****Aufrufgraphen,**

SPS Stylesheet, 326

**Aufteilen, 562****Ausblenden,**

Bereich einblenden, 531

**Ausführen,**

einzelne SQL-Anweisungen, 472

SQL, 472

SQL-Datei, 472

**Ausgabe,**

benutzerdefiniert wenn bool = false, 716

Built-in Ausführungsprozessor, 139

Daten von Ausgabe speichern, 42

Fenster, 139

Fenster - Zoom-Faktor, 139

Parameter, 716

Pseudo-SQL, 359

Schemapfad zur Ausgabe hinzufügen, 929

validieren, 142

XML validieren, 139, 156

**Ausgabe neu generieren, 359****Ausgabecodierung,**

Standard, 1146

**Ausgabesprache, 946****Ausgabeverzeichnis,**

für Codegenerierungsdateien, 1143

für XSLT-generierte Ausgabe, 1148

**Ausgabevorschau,**

Fenster, 156

Fenster - Zoomfaktor, 156

**Ausnahmeereignis,**

auslösen, 296

Webservice Fehler, 833

zu wenig Arbeitsspeicher, 1030

**Ausrichten,**

Komponenten im Mapping-Fenster, 21

**Ausrichtungshilfslinien,**

Komponenten automatisch ausrichten, 78

**Ausschneiden,**

Parent/Child-Konnektoren, 114

**Austausch,**

Dokumente - SAP IDoc, 641

**Auswählen,**

Tabellendaten - Datenbankabfrage, 483

**auto-format, 788****Autokomplettierung,**

SQL Editor, 474

**Automatisch ermitteln,**

Parameter-Datentyp, 434

**Automatisch mappen,**

Child-Elemente, 114

**Automatisch verbinden,**

Child-Elemente, 114

Sub-Einträge, 96

**Automatische Ausrichtung,**

Hilfslinien, 21

Komponenten im Mapping, 78

**Automatisches,**

Laden von Bibliotheken, 735

**auto-number, 386, 766****avg, 752**

## B

**Base Package, 1032****base-uri, 802****Basis,**

Typ - derived types, 302

**Basispaketname,**

für Java, 1121

**Bearbeiten, 922****Bedingung,**

einmal trennen, 537

erweiterbares IF-Else, 923

**Bedingung mehrmals,**

aufteilen, 540

**Beenden,**

der Applikation, 1095

**Befehlszeile,**

dynamische Input-Dateinamen, 190  
exit-Code, 266  
Input-Parameter, 193, 273  
Parameter, 266  
Parameter und Eingabewerte, 273  
Parameter und Vorschauereinstellungen, 275

**Befehlszeilenparameter,**

Wildcards in Anführungszeichen, 273

**Beginnt mit,**

mehrmals trennen, 556

**Begrenzungszeichen,**

Feld, 510

**Behandeln,**

leere Felder als nicht vorhanden, 576

**Beispiel,**

rekursives benutzerdefiniertes Mapping, 304

**Beliebig,**

einmal trennen, 559  
mehrmals trennen, 553

**Benutzerdefiniert,**

Ausgabe wenn bool = false, 716  
Bibliothek, 72, 136  
Füllzeichen (feste Länge), 522  
Funktionen - Typ ändern, 687  
Funktion, 72, 136  
Funktion - Standard, 699  
Funktionen, 687  
Funktionen - komplexe, 704  
importieren/exportieren, 687  
komplexe Input-Komponenten, 705  
lookup-Funktionen, 699  
löschen, 687  
verschachtelte Funktionen, 716  
XQuery-Funktionen, 729  
XSLT 2.0 Funktionen, 728  
XSLT-Funktionen, 724

**Benutzerdefinierte,**

Funktion - Restrictions, 687  
Funktionen & mffs, 687  
Trennzeichen, 607

**Benutzerdefinierte Bibliothek,**

hinzufügen, 733

**Benutzerdefinierte Funktion,**

inline, 697  
rekursiv, 306

**Benutzerdefinierten,**

Funktionstyp ändern, 687

**Benutzerhandbuch,**

siehe auch Bildschirmhilfe, 960

**Bestätigung,**

X12 997, 598

**Bestellen der Altova-Software, 961****Betriebssysteme für Altova-Produkte, 1292****BETWEEN,**

SQL WHERE, 434

**Beziehung,**

beibehalten/verwerfen, 417  
erstellen, 420

**Beziehungen,**

Tabellenbeziehungen hinzufügen, 420

**Bibliothek, 1049**

automatisch laden, 735  
benutzerdefiniert, 72, 136  
benutzerdefinierte Funktionen importieren, 687  
benutzerdefinierte hinzufügen, 733  
definieren, 686  
Funktion, 72, 136  
Funktion suchen, 72, 136  
Funktionsreferenz, 750  
Generatorfunktion, 386  
Komponenten-Benutzerschnittstelle definieren, 737  
neu C# 2007r3, 740  
neu C++ 2007r3, 740  
neu Java 2007r3, 740  
XPath2, 72, 136  
XQuery-Funktionen hinzufügen, 729  
XSLT 2.0-Funktionen hinzufügen, 728  
XSLT-Funktionen hinzufügen, 724

**Bibliotheken,**

und benutzerdefinierte Funktionen, 687

**Bibliotheksdatei,**

mff, 733

**Bibliotheksmodule,**

in XQuery-Dokument, 1251

**Bibliothekstyp,**

Enumerationen für C++, 1176  
für C++, 1144, 1145

**Bildschirmhilfe,**

Index, 960  
Inhaltsverzeichnis, 960  
Suchen, 960

**Blockkommentar, 475****BOM,**

**BOM,**

Byte Order Mark, 929

**Bool,**

Ausgabe wenn false, 716

Input-Nodes vergleichen, 260

**boolean, 755****Browser,**

Datenbankabfrage, 478

Filter anwenden, 479

filtern, 479

generating SQL for tables, 481

SQL generieren, 471

**Build,**

C++ Build-Konfigurationen, 978

**Build.xml, 973****Builder,**

benutzerdefinierte Funktion, 687

**Built-in,**

Ausführungsprozessor - Schaltfläche, 139

**Bytefolge-Markierung,**

in Komponenteneinstellungen, 929

# C

**C#,**

C# Webservice erstellen, 831

Code, 966

Code generieren, 975

Code kompilieren, 975

Codegenerierung, 1118

Einstellungen, 1032

Enumeration, 1179

Fehlerbehandlung, 1071

generierten Code integrieren, 987

integration of MapForce, 1194

Integration von MapForce, 1194

Optionen, 1143, 1145

**C++,**

Code, 966

Code generieren, 978

Code kompilieren, 978

Codegenerierung, 1119

EDIFACT / X12-Generierung, 625

Einstellungen, 1032

Enumeration, 1175, 1176

Fehlerbehandlung, 1071

generierten Code integrieren, 989

Konfigurationen erzeugen, 978

Optionen, 1143, 1144, 1145

**Canonical XML,**

digitale Signatur, 225

**capitalize, 796****ceiling, 770****char-from-code, 776****Child löschen,**

und aktualisieren, 375

**Child-Datenelemente,**

löschen, 114

**Child-Elemente,**

automatisch verbinden, 114

**Class ID,**

in MapForce Integration, 1186

**Close, 1117**

Projekt, 1151

**Code,**

exit-Code - Befehlszeile, 266

Generierung, 971

Generierung - Beispiel, 981

Inline-Funktionen & Codegröße, 697

integrierte Typen, 1049

MapForce Code integrieren, 984

Schemnamen daraus löschen, 929

SPL, 1034

validieren beim Ausführen von EDI-Code, 607

**code generation,**

enumerations, 1165, 1182

options for, 1146, 1147, 1148

**Code Generator, 966****Code Point,**

Collation, 236

**code-from-char, 776****Codegenerierung, 1127, 1129**

Beispiel, 1061

C#, 1118

C++, 1119

C++-Klasse - EDIFACT / X12, 625

Eingabeparameter, 273

Enumerationen, 1175, 1176, 1177, 1179, 1181

Java, 1119

Optionen, 1094, 1122, 1123, 1145, 1146, 1148

Optionen für, 1143, 1144, 1146

Pfade absolut machen, 205

und absoluter Pfad, 37

und Eingabeparameter, 273

**Codegenerierung, 1127, 1129**  
Version der Wrapper-Klasse, 950  
von verketteten Mappings, 129  
XSLT, 1120

**Codierung,**  
Standard für Ausgabedateien, 1146

**Collation,**  
Locale Collation, 236  
Sortierkomponente, 236  
Unicode Code Point, 236

**Collations,**  
in XPath 2.0, 1254  
in XQuery-Dokument, 1251

**columnname-to-index, 801**

**COM-API,**  
Dokumentation, 1058

**compare, 809**

**Compiler,**  
Einstellungen, 1032

**ComplexType,**  
sortieren, 236

**Component, 1106**

**Components, 1112**

**Compute when,**  
Variable, 216

**concat, 776**

**Config,**  
für mehrere Nachrichten aktualisieren, 639

**Connection, 1114**

**constructors,**  
xs:ENTITY, 803

**Container,**  
aus-/einblenden, 531  
Ausgabe FlexText ignorieren, 568

**contains, 776**

**Context,**  
override, 285

**Context menu,**  
for tables Database Query, 481

**convert-to-utc, 790**

**Copyright-Informationen, 1298**

**Core,**  
Bibliotheksfunktionen, 752

**cos, 794**

**count-substring, 796**

**Count, 752, 1102, 1112, 1125, 1141**

**count() function in XPath 2.0,**  
siehe fn:count(), 1254

**count()-Funktion,**  
in XPath 1.0, 1246

**count, sum, avg,**  
Aggregatfunktion, 255

**CR / LF,**  
in Datenbanken ersetzen, 253

**create-guid, 794**

**CSV,**  
Dateioptionen, 510  
Feld "Benutzerdefiniert", 510  
Hierarchien erstellen - Schlüssel, 506  
Input-Datei, 510  
mappen, 499  
Output-Datei, 510  
speichern als, 570  
Streaming, 139

**CSV. Feld-Datentypen, 510**

**current, 814**

**current-date, 804**

**current-dateTime, 804**

**current-time, 804**

## D

**Database Query,**  
context menu options, 481  
text font options, 489

**Datapoint, 1115**

**Date,**  
XSLT 2.0 Konstruktor, 728

**date-from-datetime, 790**

**Datei, 917**  
Encoding, 510, 522  
globale Ressourcen definieren, 160  
Input aus mehreren Dateien / Output in mehrere Dateien, 55  
Katalog, 298  
Ressourcenkonfiguration hinzufügen, 160

**Datei mit fester Länge,**  
leere erstellen, 522

**Datei:,**  
(Standard), 55  
Datenelement in Komponente, 55

**Dateien,**  
dynamische Dateinamen als Input, 193  
mehrere Dateien anhand einer Datenbank, 195  
mehrere von Datenbank, 198



**Dateien,**

mehrere von Excel, 200

**Daten,**

DSN, 462

filtern, 32

Unterstützung von Streams, 991

**Daten beibehalten,**

bei Verwendung von Wertezuordnung, 248

über Wertezuordnung übergeben, 248

**Daten konsolidieren,**

XML-Dateien zusammenführen, 264

**Daten sortieren,**

Sortierkomponente, 236

**Daten übergeben,**

über Wertezuordnung unverändert übergeben, 248

**Daten unverändert beibehalten,**

über Wertezuordnung übergeben, 248

**Datenbank,**

aktualisieren, 469

als globale Ressource, 178

Änderungen rückgängig machen, 469

auf XBRL (Mapping), 871

Beziehung erstellen, 420

Daten mappen - Java, 63

Datenbank ändern, 350

Datenbankabfrage (Register), 460

DB ändern, 929

einfügen, 63

Funktionsmatrix, 404

große DB mappen, 424

IBM DB2 Info, 415

JDBC-Verbindungen, 394

komplette Mappen, 425

mappen auf, 349

mappen von XML, 350

mehrere Quellen, 929

mehrere XML-Dateien generieren, 198

MS Access Info, 405

MS SQL Server, 407

MySQL Info, 411

Oracle Info, 409

Schema zuweisen, 438

Schemanamen aus Code entfernen, 929

Sonderzeichen ersetzen, 253

Sybase Info, 413

Tabellenaktionen, 398

Tabellenvorschau, 455

Tabellenvorschau anzeigen, 442

teilweises Mapping, 426

Unterstützung, 338

Verbindungsassistent, 462

XML-Daten mappen - generische Methode, 438

XQuery, 338

**Datenbankabfrage,**

Autokomplettierung, 474

Lesezeichen, 476

Optionen - Kodierung, 486

Optionen Ergebnisansicht, 489

Regionen, 476

SQL ausführen, 472

SQL Editor-Funktionen, 473

SQL generieren, 471, 478, 483

SQL-Fenster, 470

Tabellen filtern, 479

Text auskommentieren, 475

**Datenbankabfrage (Register), 460****Datenbankaktion,**

Aktualisieren, 367

Child aktualisieren und löschen, 375

einfügen, 359

ignorieren, 383

löschen, 379

**Datenbankbeziehungen,**

beibehalten/verwerfen, 417

**Datenbankfunktionen,**

set-null, 428

**Datenbankunterstützung, 59****Datenelement,**

duplizieren, 49

fehlendes, 118

Rows, 503

**Datenelemente,**

für Excel 2007 definieren, 659

Schema - Mapping, 25

zusammengeführte Einträge aufgliedern, 636

**Datenquelle,**

FlexText, 548

**Datensatz,**

XML von Datensatz generieren, 198

**Datensätze löschen,**

vor Tabellenaktion, 398

**Datentyp,**

SQL WHERE automatisch ermitteln, 434

**Datentyp,**

explizit - implizit, 728

**Datentypen,**

**Datentypen,**

Feld, 510

in XPath 2.0 und XQuery 1.0, 1254

**Datetime,**

Mappen mit Hilfe von Funktionen, 607

**datetime-add, 790****datetime-diff, 790****datetime-from-date-and-time, 790****datetime-from-parts, 790****datetime-to-xml, 801****date-to-xml, 801****Datum,**

xsd:date, 607

**day-from-datetime, 790****day-from-duration, 790****DB,**

ORDER BY, 430

**DB Query,**

autocomplete options, 488

saving / opening SQL, 473

text font options, 487

**DB2,**

als Zielkomponente, 450

MS Access auf IBM DB2 mappen, 452

Vorschau von XML-Inhalt anzeigen, 442

XML-Daten abfragen, 448

XML-Daten mappen, 442

**deep-equal() function in XPath 2.0,**

siehe fn:deep-equal(), 1254

**Default,**

Eingabewert, 716

**Default Functions Namespace,**

für XPath 2.0- und XQuery 1.0-Ausdrücke, 1254

in XSLT 2.0 Stylesheets, 1248

**default-collation, 804****Definitionsdatei,**

globalresource.xml, 158

**degrees, 794****Derived,**

types - verwenden / mappen auf, 302

**Detached,**

digitale Signatur, 225

**Diagrammfunktionen,**

Beispiel, 1287

Diagrammdatenstruktur, 1282

Liste, 1279

**Digitale Signatur,**

aktivieren, 220

Einstellungen, 225

**Dimension,**

in Komponente anzeigen, 859

Standardwerte - identifizieren und period, 859

value-map-Transformation, 865

verschieben, 859

**distinct-values, 775****divide, 770****divide-integer, 794****DII,**

Compiler-Einstellungen, 1032

**document, 814****document-uri, 802****Dokument, 1116**

aktives Dokument abrufen, 1125

beim Öffnen, 1090

beim Schließen, 1117

Dateiname, 1122

gespeichert, 1124

Mapping, 320

neues erstellen, 1093, 1126

öffnen, 1094, 1126

Pfad und Name, 1118

Pfad zu, 1123

schließen, 1117

speichern, 1123

speichern unter, 1124

**Dokumentation,**

SPS Stylesheets definieren, 329

**Dokumente, 1091, 1125**

abrufen, 1126

Gesamtzahl in Sammlung, 1125

**Dokumentebene,**

Beispiele für die Integration von MapForce, 1194

Integration des MapForceControl, 1189

Integration von MapForce, 1190, 1191, 1192, 1193

**Domain,**

im Komponente anzeigen, 859

**DOM-Typ,**

Enumerationen für C++, 1175

für C++, 1144

**DoTransform bat,**

XML transformieren, 42

**Drag and Drop,**

SQL-Anweisung generieren, 471

**Drucken,**

nicht druckbare Trennzeichen, 607

**DSN,**

**DSN,**

- Data Source Name, 462
- Verbindungsassistent, 462

**DTD,**

- Quelle und Ziel, 294

**Duplikat einfügen,**

- vorher oder danach einfügen, 929

**Duplizieren,**

- Eingabeelement, 49
- Konnektor mittels Strg duplizieren, 96

**duration-add, 790****duration-from-parts, 790****duration-subtract, 790****Dynamisch,**

- Dateinamen als Input-Parameter, 193
- Input-Dateien zur Laufzeit, 190
- und Unterstützung mehrerer Dateien, 188

# E

**Eclipse,**

- install MapForce plugin, 891
- MapForce examples importieren, 902
- MapForce Nature anwenden, 910
- MapForce Plug-in, 884
- MapForce Plugin installieren, 886, 900
- MapForce Plugin starten, 897
- neues Mapping / Projekt erstellen, 904

**EDI,**

- als Zielkomponenten, 607
- EDIFACT Autokomplettierungsregeln, 611
- EDIFACT Validierungsregeln, 611
- IATA PADIS, 643
- Komponenteneinstellungen, 607
- SAP IDocs, 641
- Trennzeichen, 584
- Trennzeichenrangfolge, 607
- Upgrade für mehrere Nachrichten, 639
- validieren, 142
- X12 Autokomplettierungsregeln, 614
- X12 HIPAA, 646
- X12 TA1 Segment, 602
- X12 Validierungsregeln, 614
- zusammengeführte Einträge aufgliedern, 636

**edifact, 582, 788**

- als Ziel, 607

angepasste Beispieldatei, 625

anpassen, 617

anpassen - Konfiguration, 625

anpassen - vorbereiten, 618

auf XML mappen, 585

Autokomplettierungsregeln, 611

Codegenerierung - C++-Klasse, 625

config-Datei für mehrere Nachrichten aktualisieren, 639

globale Anpassung, 621

inline-Anpassung, 623

Terminologie, 584

Validierungsregeln - Ziel, 611

zusammengeführte Einträge, 628

**Edition, 1091****Eigenschaften,**

- Wertezuordnungstabelle, 251

**Eine einzige Zieldatei,**

- mehrere Quellen, 264

**Einfügen, 923**

- Blockkommentar, 475
- Datenbank, 63
- Datenbankaktion, 359
- Kommentare, 475
- Lesezeichen, 476
- Regionen, 476
- SQL WHERE Operator, 434
- SQL WHERE-Komponente, 430
- Webservice-Aufruf, 837, 842
- Zeilenkommentar, 475

**Einführung,**

- Code Generator, 967
- in FlexText, 530

**Einführung in MapForce, 3****Eingabemethoden, 1295****Eingabeparameter,**

- Befehlszeile, 273
- und Codegenerierung, 273

**Einmal berechnen,**

- Variable, 216

**Einmal trennen, 558**

- beliebig, 559
- feste Länge, 558
- zeilenbasiert, 560

**Einstellungen,**

- Alle Concepts XBRL, 851
- allgemeine, 486
- C++ und C#, 1032
- digitale Signatur, 225

**Einstellungen,**

- Ergebnisansicht -- Datenbankabfrage, 489
- Füllzeichen, 522
- Komponente ändern, 929
- Komponentenname, 514, 929
- Textdatei mit fester Länge, 522
- WSDL-Aufrufseinstellungen, 837

**Eintrag,**

- zusammengeführte Einträge aufgliedern, 636

**Element,**

- in Zielelement konvertieren, 929
- rekursives Element in XML-Schema, 304

**element-available, 814****empty, 796****Encoding,**

- Datei, 510, 522
- default for output files, 1146
- in XQuery-Dokument, 1251

**Endbenutzer-Lizenzvereinbarung, 1298, 1302****ends-with, 809****Entfernen,**

- "Alles kopieren"-Verbindungen, 109
- Blockkommentar, 475
- Kommentare, 475
- Lesezeichen, 476
- Regionen, 476
- Schemanamen, 929
- Tabellen aus Komponente, 350
- Verbindung, 114
- Zeilenkommentar, 475

**Enumerationen, 1164, 1175, 1176, 1177, 1179**

- für MapForce Ansicht, 1181
- in MapForceControl, 1241

**Enumerations, 1165, 1166, 1167, 1168, 1171, 1172, 1173, 1182****Enveloped,**

- digitale Signatur, 225

**equal, 768****equal-or-greater, 768****equal-or-less, 768****Ergebnisse,**

- Fenster - Datenbankabfrage, 483
- Symbole in Datenbankabfrage, 483

**Errorlevel,**

- Befehlszeilenausführung, 266

**ErrorMarkers, 1127, 1129****Ersetzen,**

- fehlenden Node, 281

**Erste Zeile,**

- als Spaltenüberschrift, 659

**Erstellen,**

- benutzerdefinierte Funktion, 687
- Funktion, 28
- neues Mapping / Projekt in Eclipse, 904
- Regionen, 476
- Webservice-Projekt, 822

**Erweitern,**

- Bereich reduzieren, 531
- Funktionsparameter, 72, 136
- Regionen, 476

**Erweiterungsfunktionen für XSLT und XQuery, 1259****Erweiterungsfunktionen in .NET für XSLT und XQuery,**

- siehe .NET Erweiterungsfunktionen, 1267

**Erweiterungsfunktionen in Java für XSLT und XQuery,**

- siehe Java-Erweiterungsfunktionen, 1259

**Erweiterungsfunktionen in MSXSL Scripts, 1273****escape-uri, 809****Evaluierungs-Keycode,**

- für Ihre Altova-Software, 961

**Evaluierungszeitraum,**

- bei Altova Software-Produkten, 1301
- für Altova-Software-Produkte, 1299
- von Altova Software-Produkten, 1298

**Events,**

- des Dokuments, 1117

**Events für Projekt, 1149****Examples,**

- Tutorial-Ordner, 18

**Excel,**

- auf XBRL - Beispiel, 867
- erste Zeile als Spaltenüberschrift, 659
- mehrere XML-Dateien generieren, 200
- Nodes duplizieren, 659

**Excel 2007,**

- Arbeitsblatt - Datenelemente definieren, 659
- Arbeitsmappe - Datenelemente definieren, 659
- auf XML mappen, 678
- Datenbank mappen auf, 681
- mapbare Datenelemente- definieren, 659
- OOXML Excel 2007, 658

**Exist,**

- Mappen fehlender Nodes mittels not-exist, 283

**exists, 772**

- Node-Test, 281

**exit-Code, 266****exp, 794**

**Explizit,**

Datentyp, 728

**Explizite,**

Beziehungen - lokale Beziehungen, 420

**Exportieren,**

benutzerdefinierte Funktion, 687

**Expression,**

regular, 784

**Externe Funktionen,**

in XQuery-Dokument, 1251

**Extras, 950**

## F

**fadians, 794****false, 803****Farbverlauf,**

im Hintergrund, 1148

**Fault,**

Webservice-Aufruf, 837

**Fehlend,**

leere Felder, 510, 522

**Fehlende Datenelemente, 118****Fehlende Felder,**

als nicht vorhanden behandeln, 576

**Fehlende Nodes,**

Mappen fehlender Nodes, 283

**Fehler,**

Definieren von Ausnahmeereignissen, 296

Validierung, 142

Webservice, 833

**Fehlerbehandlung,**

allgemeine Beschreibung, 1071

**Feld,**

Begrenzungszeichen, 510

benutzerdefiniert - CSV, 510, 522

feste Länge, 514

Schlüssel in Textdateien, 506

**Felder,**

in Tabellenaktionen vergleichen, 400

**Felder mit fester Länge,**

Feld "Benutzerdefiniert", 522

**Feldtyp überprüfen, 514****Feste Länge,**

Dateien mappen, 498

einmal trennen, 558

Input-Datei, 522

mappen, 514

mehrmals trennen, 553

Output-Datei, 522

Textdateieinstellungen, 522

**Filter,**

"Alles kopieren"-Konnektor, 109

Daten, 32

Datenbankobjekte, 479

den Online Browser, 479

Komplement, 32

Komponente - Tipps, 234

Parent-Datenelemente mappen, 234

Prioritätskontext, 234

verketteten - vermeiden, 234

**Filtern,**

Zusammenführen von XML-Dateien, 264

**find-substring, 796****Flat File,**

mappen, 498

**Flat Format,**

mappen, 514

**FlexText,**

als Zielkomponente, 549

Datenquelle, 548

Einführung, 530

Tutorial, 534

Vorlage in MapForce, 545

**FLF,**

speichern als, 576

Streaming, 139

**floor, 770****fn:base-uri in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:collection in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:count() in XPath 2.0,**

und Whitespace, 1254

**fn:current-date in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:current-dateTime in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:current-time in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:data in XPath 2.0,**

Unterstützung in Altova-Prozessoren, 1255

**fn:deep-equal() in XPath 2.0,**

und Whitespace, 1254

- fn:id in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:idref in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:index-of in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:in-scope-prefixes in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:last() in XPath 2.0,**
    - und Whitespace, 1254
  - fn:lower-case in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:normalize-unicode in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:position() in XPath 2.0,**
    - und Whitespace, 1254
  - fn:resolve-uri in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:static-base-uri in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - fn:upper-case in XPath 2.0,**
    - Unterstützung in Altova-Prozessoren, 1255
  - format-date-time, 755**
  - format-guid-string, 796**
  - format-number, 755, 814**
  - Fragen und Antworten zu MapForce, 963**
  - Fragezeichen,**
    - fehlende Datenelemente, 118
  - From, 805**
  - FullName, 1118, 1152**
  - Füllzeichen,**
    - Datei mit feste Länge, 522
    - entfernen, 514, 522
  - Functional Acknowledgement,**
    - X12 997, 598
  - function-available, 814**
  - functions,**
    - XPath 2.0 and XQuery 1.0, 1254
  - Funktion, 766, 788, 800, 943**
    - Aggregatfunktion, 255
    - benutzerdefiniert, 687
    - benutzerdefinierte, 72, 136
    - benutzerdefinierte - Typ ändern, 687
    - benutzerdefinierte exportieren, 687
    - benutzerdefinierte Funktion, 306
    - benutzerdefinierte Standardfunktion, 699
    - benutzerdefinierte XQuery-Funktion hinzufügen, 729
    - benutzerdefinierte XSLT 2.0-Funktion hinzufügen, 728
    - benutzerdefinierte XSLT-Funktion hinzufügen, 724
    - Bibliothek, 72, 136
    - core Bibliothek, 752
    - Datenbank set-null, 428
    - Datetime, 607
    - definieren, 686
    - erweiterbare, 72, 136
    - erweiterbares IF-Else, 923
    - Generator, 386
    - hinzufügen, 72, 136
    - Implementierung, 739
    - in Bibliothek suchen, 72, 136
    - inline, 697
    - Input als Parameter, 273
    - komplexe - inline, 697
    - Konvertierung - Boolesche, 260
    - lang Bibliothek, 790
    - lookup-Funktion, 699
    - Restrictions in benutzerdefinierter, 687
    - summieren, 730
    - Typ von benutzerdefinierter Funktion ändern, 687
    - verschachtelte benutzerdefinierte, 716
    - visueller Function Builder, 687
    - xlsx-Bibliothek, 801
    - xmlexists - DB2 abfragen, 448
    - xpath2 Bibliothek, 802
    - XQuery, 72, 136
    - xslt-Bibliothek, 812
  - Funktionen,**
    - benutzerdefinierte importieren, 687
    - mappen auf, 28
    - Referenzteil, 750
    - siehe unter XSLT 2.0-Funktionen, 1250
  - Funktionsbestätigung,**
    - X12 997, 598
  - Für alle ausgelösten Bedingungen, 562**
  - Für die erste ausgelöste Bedingung, 562**
- ## G
- Gemappter Wert,**
    - Schlüsseleinstellungen - Tabellenaktionen, 398
  - generate-id, 814**
  - Generator,**
    - Funktion, 386
  - Generieren,**

**Generieren,**

- C#-Code, 975
- C++-Code, 978
- Code, 971
- Code - Beispiel, 981
- Code & Inline-Funktionen, 697
- Code von Schema, 966
- digitale Signatur, 220
- Java-Code, 973
- mehrere Java-Ziele, 42
- mehrere XSLT-Ziele, 42
- SQL, 471
- XML-Schema automatisch generieren, 20

**Gesamtdokumentation,**

- SPS Stylesheet, 326

**get-fileext, 764****get-folder, 764****GetRootDatapoint, 1170****Global, 1295****Globale Anpassung,**

- EDIFACT, 621
- X12, 632

**Globale Objekte,**

- in SPL, 1039

**Globale Ressource,**

- aktivieren, 165
- Arbeitsablauf, 171
- Arbeitsablauf starten, 175
- Datenbank als, 178
- Eigenschaften, 183
- einer Komponente zuweisen, 163
- Konfiguration kopieren, 160
- Ordner als, 167
- Standardkonfiguration, 160

**Globale Ressourcen,**

- Definitionsdatei, 158
- Ressourcendatei definieren, 160
- Symbolleiste, 159

**Globalresource.xml,**

- Ressourcendefinition, 158

**greater, 768****Größe anpassen,**

- Komponente an optimale Größe, 20

**Große Datenbank,**

- importieren, 425

**group-adjacent, 775****group-by, 775****group-ending-with, 775****group-starting-with, 775****Gruppen,**

- Schleifen iterieren, 503
- Schleifen und Hierarchien, 145

**guid, 386**

## H

**Health Level 7,**

- Beispiel, 654

**Hebräisch, 1296****Hierarchie,**

- Node in FlexText, 566
- Tabelle, 357
- von Textdateien, 506

**Hierarchien,**

- Schleifen und Gruppen, 145

**HighlightMyConnections, 1132****HighlightMyConnectionsRecursively, 1132****Hilfe, 959**

- siehe Bildschirmhilfe, 960

**Hintergrund,**

- mit Farbverlauf, 1148
- schattiert, 1032

**Hintergrundinformationen, 1291****Hinzufügen,**

- benutzerdefinierte Bibliothek, 733
- benutzerdefinierte Funktionen, 687
- Beziehungen zu Tabellen, 420
- Duplikat davor oder danach einfügen, 929
- globale Ressourcendatei, 160
- Schemapfad, 929

**HIPAA,**

- X12, 646

**Hiragana, 1295****HL7, 582**

- Unterkomponenten-Trennzeichen, 607

**HL7 2.6 auf 3.x,**

- Beispiel, 654

**hour-from-datetime, 790****hour-from-duration, 790****HRESULT,**

- und Fehlerbehandlung, 1071

**HTML,**

- Integration von MapForce, 1202
- Mapping-Dokumentation, 320

**HTML,**

- Register im Mapping-Fenster, 68
- Stylevision Power Stylesheets, 314

**HTML-Beispiel,**

- für MapForceControl Integration, 1186, 1187

**Hypercube,**

- XBRL-Domains/Dimensions, 859

**I****IATA PADIS, 643****IBM DB2,**

- als Zielkomponente, 450
- Daten von der Quelldatenbank mappen, 452
- Datenbankinfo, 415
- XML-Daten abfragen, 448
- XML-Daten mappen, 442
- XML-Schema in Komponente einbetten, 442

**IDocs,**

- SAP-Dokumentaustauschformat, 641

**IF-Else,**

- erweiterbar, 923

**Ignorieren, 568**

- Datenbankaktion, 383
- Untergeordnete Input-Daten, 400

**Implementation Acknowledgement,**

- X12 999, 600

**Implementierung,**

- Funktion, 739

**Implementierungsspezifisches Verhalten,**

- von XSLT 2.0-Funktionen, 1250

**Implizit,**

- Datentyp, 728

**Implizite Zeitzone,**

- und XPath 2.0-Funktionen, 1254

**Importieren,**

- benutzerdefinierte Funktionen, 687

**IN,**

- SQL WHERE, 434

**Include,**

- XSLT, 724
- XSLT 2.0, 728

**in-context,**

- Parameter, 752

**index-to-columnname, 801****Inhalt beginnt mit, 562****Inhalt enthält, 562****Inline, 687**

- benutzerdefinierte Funktionen, 697
- Funktionen und Codegröße, 697

**Inline-Anpassung,**

- EDIFACT, 623

**Input,**

- als Befehlszeilenparameter, 273
- Befehlszeilenparameter, 275
- Boolesche Nodes vergleichen, 260
- Datei - CSV, 510
- Datei - Text, 522
- mehrere Dateien, 190
- optionale Parameter, 716
- Standardwert, 716
- XML-Instanz, 929

**Input-Komponente,**

- Standardwert, 275

**Input-Parameter,**

- dynamisch, 190
- und dynamische Dateinamen, 193

**Input-Symbol,**

- Mapping, 96

**Input-Verarbeitung aktivieren,**

- Optimierung, 929

**InsertXMLFile, 1133****InsertXMLSchema, 1133****InsertXMLSchemaWithSample, 1133****Install,**

- plug-in for Eclipse, 891

**Installation,**

- Ordner examples, 18

**Installieren,**

- Plug-in für Eclipse, 886

**Instanz,**

- obligatorische XBRL Items, 868
- XML-Input-Instanz, 929
- XML-Output-Instanz, 929

**Integrating,**

- MapForce in Applikationen, 1184

**Integrieren,**

- in C#, 987
- in C++, 989
- in Java, 985

**Integrieren MapForce Code, 984****Internet-Verwendung,**

- in Altova-Produkten, 1297

**iSeries,**



**iSeries,**

timeout deaktivieren, 442

**ISO,**

8601 Datums- und Uhrzeitformat, 607

**ISO/IEC 10646, 1295****is-xsi-nil, 772****Item, 1102, 1112, 1126, 1141**

XBRL Standardwerte, 851

**Iteration,**

Prioritätskontext, 262

**Iterieren,**

durch Textdateien, 503

## J

**Java,**

Code, 966

Code generieren, 973

Codegenerierung, 1119

Datenbankdaten mappen, 63

Einstellungen, 1032

generierten Code integrieren, 985

Java-Webservice erstellen, 825

mehrere Ziele, 37

mehrere Ziele generieren, 42

Optionen, 1121, 1143

options, 1147

**Java Erweiterungsfunktionen,**

Datentypkonvertierungen, Java in XPath/XQuery, 1267

Instanzmethoden, Instanzfelder, 1265

Konstrukturen, 1264

statische Methoden, statische Felder, 1264

**Java extension functions,**

datatype conversions, XPath/XQuery to Java, 1266

**Java-Erweiterungsfunktionen,**

benutzerdefinierte JAR-Dateien, 1263

benutzerdefinierte Klassendateien, 1261

für XSLT und XQuery, 1259

Übersicht, 1259

**JavaScript,**

Fehlerbehandlung, 1071

**JDBC-Verbindungen, 394****JScript,**

Beispiel für Codegenerierung, 1061

## K

**Kanji, 1295****Katakana, 1295****Katalogdateien, 298****Keycodes,**

für Ihre Altova-Software, 961

**Kodierung,**

Bytefolge-Markierung, 929

Datenbankabfrageoptionen, 486

Komponenteneinstellungen, 929

**Komma,**

CSV-Dateien, 499

**Kompatibilitätsmodus,**

v2005R3, 1032

**Kompilieren,**

C#-Code, 975

C++-Code, 978

**Komplement,**

Komplement Nodeset, 32

**Komplex,**

benutzerdefinierte Funktion, 704

komplexe benutzerdefinierte Input-Komponenten, 705

**Komplexe,**

Funktion - inline, 697

**Komponente,**

Ausnahmeereignis, 296

Daten sortieren, 236

Datenbank, 59

Datenbank ändern, 929

Domain und Dimensions, 859

EDI-Einstellungen, 607

Einstellung Alle Concepts, 851

Einstellungen ändern, 929

FlexText in MapForce verwenden, 545

gelöschte Datenelemente, 118

globale Ressource zuweisen, 163

Input - Standardwert, 275

Input aus mehreren Dateien / Output in mehrere Dateien, 54

Input-Verarbeitung aktivieren, 929

Kodierungseinstellungen, 929

mehrere Input- / Output-Komponenten, 190

mehrere Inputs / Outputs, 188

Name - Komponenteneinstellungen, 514, 929

optimale Größe, 20

**Komponente,**

- Pretty-Print-Ausgabe, 929
- Schema in DB2 einbetten, 442
- SPS zuweisen, 316
- SQL-Server Schema zuweisen, 455
- Tabellen hinzufügen/entfernen, 350
- UI definieren, 737
- Unterkomponenten-Trennzeichen, 607
- XBRL-Komponenteninhalt, 851

**Komponenten,**

- Input aus mehreren Dateien / Output in mehrere Dateien, 55
- Verarbeitungsreihenfolge, 152

**Komponenten-Download-Center,**

- auf der Altova Website, 963

**Komponenteneinstellungen,**

- Komponentenname, 514, 929

**Konfiguration,**

- bestehende kopieren, 160
- wechseln - globale Ressource, 165
- zu globaler Ressource hinzufügen, 160

**Konfigurieren,**

- mff-Datei, 735
- SQL Editor Einstellungen, 486

**Konnektor,**

- Eigenschaften, 114
- Mappen mit, 96
- mit Strg kopieren, 96
- Popup, 96
- untergeordnete Verbindungen inkludieren, 114

**Konnektoren,**

- Alles kopieren, 109

**Konnektor-Symbol,**

- Popup, 96

**Konstante,**

- als Standardwert, 275

**Konstruktor,**

- XSLT2, 728

**Konstruktoren,**

- xs:ENTITY, 803

**Kontext,**

- Priorität, 72, 136
- Prioritätskontext, 262

**Konvertieren,**

- in Zielschema, 929

**Konvertierung,**

- Funktionen - Boolesche, 260
- Typüberprüfung, 295

**Kopieren,**

- bestehenden Konnektor mit Strg an eine andere Stelle kopieren, 96

**L****lang, 807, 812**

- Bibliotheksfunktionen, 790

**last, 804, 812****last() function in XPath 2.0,**

- siehe fn:last(), 1254

**last()-Funktion,**

- in XPath 1.0, 1246

**Layout,**

- Browser, 478

**leapyear, 790****Leere,**

- Felder als nicht vorhanden behandeln, 576
- neue Textdatei erstellen, 522

**Leere Felder,**

- als fehlend behandeln, 510
- als nicht vorhanden behandeln, 522

**left, 796****left-trim, 796****Lesezeichen,**

- einfügen, 476
- entfernen, 476
- Lesezeichenleiste, 476
- navigieren, 476

**less, 768****Lib,**

- Compiler-Einstellungen, 1032

**LIKE,**

- SQL WHERE, 434

**Lizenz, 1302**

- für Ihre Altova-Software, 961
- Informationen, 1298

**Lizenzüberwachung,**

- bei Altova-Produkten, 1300

**Locale Collation, 236****local-name, 807, 812****local-name-from-QName, 790****log, 794****log10, 794****logical-and, 768****logical-not, 768****logical-or, 768**

**logical-xor, 794****Logo,**

- Anzeige bei Programmstart, 1147
- Druckoption, 1147

**Lokale,**

- Beziehungen, 420
- Schemas - Katalogdateien, 298

**Lookup-Tabelle,**

- Eigenschaften, 251
- Mappen fehlender Nodes, 283
- value-map Dimensions, 865
- Wertezuordnungstabelle, 244

**Löschen,**

- "Alles kopieren" Verbindungen, 109
- benutzerdefinierte Funktionen, 687
- Daten in untergeordneten Tabellen, 400
- Datenbankaktion, 379
- Löschung - fehlende Datenelemente, 118
- Tabellen aus Komponente, 350
- Verbindungen, 114

**lowercase, 796****lower-case, 809**

# M

**main-mfd-filepath, 764****manespace-uri, 807****MapForce,**

- API, 1058
- Einführung, 3
- Integration, 1184
- Parent, 1134
- Plug-in für Eclipse, 884
- Plug-in für VS .NET, 878
- Prozessor, 139, 156
- Terminologie, 12
- Übersicht, 10

**MapForce Ansicht,**

- Enumerationen für, 1181

**MapForce API, 1058**

- aufrufen, 1221
- Übersicht, 1059

**MapForce API Typbibliothek, 1088****MapForce Befehlstabelle, 1207****MapForce Integration,**

- Beispiel, 1186, 1187

- und Clients, 1184, 1189

**MapForce Plug-in,**

- Anwenden der MapForce Nature, 910
- Editor, Ansicht, Perspektive, 900
- neues Mapping / Projekt erstellen, 904
- Ordner examples importieren, 902

**MapForceCommand,**

- in MapForceControl, 1223

**MapForceCommands,**

- in MapForceControl, 1225

**MapForceControl, 1226**

- Beispiel für die Integration auf Applikationsebene, 1186
- Beispiel für Integration auf Applikationsebene, 1186, 1187
- Beispiele für die Integration auf Dokumentebene, 1194
- Dokumentation, 1184
- Integration auf Applikationsebene, 1185
- Integration auf Dokumentebene, 1189, 1190, 1191, 1192, 1193
- Integration mittels C#, 1194
- Integration mittels HTML, 1202
- Integration mittels Visual Basic, 1206
- integration using C#, 1194
- object reference, 1222

**MapForceControlDocument, 1232****MapForceControlPlaceHolder, 1238****MapForce-Prozessor,**

- Built-in Ausführungsprozessor, 139

**MapForceView, 1122, 1131**

- Applikation, 1132

**Mappen,**

- Arbeitsblätter, 659
- auf Datenelement Rows, 503
- CSV-Dateien, 499
- Daten auf Datenbanken, 349
- Datenbankdaten - Java, 63
- Excell 2007-Datenelemente definieren, 659
- Flat File Format, 498
- große Datenbank - Teil, 426
- große Datenbanken, 424
- große Datenbanken - komplett, 425
- Textdateien, 514
- und Abfragen von XML-Daten, 448
- von/auf Datenstreams, 991
- XML-Daten - generische Methode, 438

**Mapping,**

- Anzahl der Verbindungen, 96
- Child-Elemente, 114
- Dokument, 320

**Mapping,**

- Eigenschaften, 114
- Fenster - automatische Ausrichtung, 21
- Konnektor, 96
- quellorientiert - Mixed Content, 101
- Schemaelemente, 25
- Standard-Mapping, 107
- Struktur validieren, 142
- Tutorial, 18
- typorientiert, 109
- Validierung, 142
- Verarbeitungsreihenfolge, 152
- vordefinierte SPS Stylesheets für die Dokumentation, 326
- XBRL-Dateien, 846
- XML-Datei einfügen, 1133
- XML-Schema-Datei einfügen, 1133
- zielorientiert, 107

**Mapping methods,**

- standard, 100
- target-driven, 100

**Mapping-Fenster,**

- automatische Ausrichtung, 78
- Stylevision-Register, 68

**Markierte Datenelemente,**

- fehlende Datenelemente, 118

**matches, 809****match-pattern, 796****max, 752, 794****Mehrere,**

- Bedingungen - aufteilen, 562
- mehrere Zielschemas anzeigen, 42
- Quellen und Codegenerierung, 981
- Quellschemas, 929
- Tabellenaktionen, 367
- Ziele und Codegenerierung, 981
- Ziele und Java, 37
- Zielschemas, 37

**Mehrere aufeinander folgende Elemente,**

- Edifact - X12, 628

**Mehrere Dateien,**

- Input / Output, 190
- Input aus / Output in, 54
- verarbeiten (Tutorial), 55

**Mehrere Inputs,**

- Elemente, 49

**Mehrere Nachrichten,**

- EDI config aktualisieren, 639

**Mehrere Quellen,**

- auf ein einziges Zielelement, 44
- in einer einzigen Zieldatei zusammenführen, 264

**Mehrere Tabellen,**

- auf eine XML-Datei, 257

**Mehrere XML-Dateien,**

- anhand einer einzigen XML-Quelle, 195
- von einer Datenbank, 198
- von einer Excel-Datei, 200

**Mehrmals,**

- trennen - aufteilen (Bedingung), 540

**Mehrmals trennen,**

- festе Länge, 553
- getrennt (beginnt mit), 556
- getrennt (beliebig), 553
- getrennt (zeilenbasiert), 555
- Standard, 552

**Meldungen,**

- Fenster - Datenbankabfrage, 483
- Symbole in Datenbankabfrage, 483

**Menü,**

- Ansicht, 948
- Ausgabesprache, 946
- Bearbeiten, 922
- Datei, 917
- Einfügen, 923
- Extras, 950
- Funktion, 943
- Hilfe, 959
- Verbindung, 938

**Methode,**

- Namen reservieren, 1030

**MFC-Unterstützung,**

- von C++, 1145

**mfd-filepath, 764****mff,**

- Bibliotheksdatei, 733
- mff.xsd-Datei, 733
- und benutzerdefinierte Funktionen, 687

**mff-Datei,**

- konfigurieren, 735

**Microsoft® Office 2007,**

- mappen von/auf, 658

**millisecond-from-duration, 790****min, 752, 794****min, max,**

- Aggregatfunktion, 255

**minOccurs/maxOccurs,**

- Optimierung der Input-Verarbeitung, 929

**minute-from-datetime, 790**

**minute-from-duration, 790**

**missisecond-from-datetime, 790**

**Mixed,**

Content mappen, 101

quellorientiertes Mapping, 101

Standard-Mapping, 107

**modulus, 770**

**Modus,**

Kompatibilität mit v2005R3, 1032

**month-from-datetime, 790**

**month-from-duration, 790**

**MS Access,**

Datenbankinfo, 405

Unterstützung v 2000 und 2003, 338

**MS Access 2000,**

Unterstützung, 59

**MS SQL Server,**

Datenbankinfo, 407

Unterstützung, 59

**MS Visual Studio .NET,**

MapForce Plug-in, 878

**MSXML 6.0,**

Bibliothek, 950

**msxsl:Script, 1273**

**Multi,**

Dateiunterstützung - Sprachen, 188

Input / Output, 188

**multiply, 770**

**MyDokuments,**

Ordner examples, 18

**MySQL,**

Datenbankinfo, 411

## N

**Nach oben verschieben,**

Datenelement in Komponente, 929

**Nach unten verschieben,**

Datenelement in Komponente, 929

**Nachricht,**

config-Datei für mehrere Nachrichten aktualisieren, 639

**Name, 807, 812, 1093, 1122, 1154**

**Named,**

Template - Namespaces, 724

**Named Template,**

Nodes summieren, 730

**Namespace,**

named Template, 724

**Namespace URI,**

DTD, 294

**Namespaces,**

im XSLT 2.0 Stylesheet, 1248

in XQuery-Dokument, 1251

**namespace-uri, 812**

**namespace-uri-from-QName, 790**

**Native,**

Verbindung, 387

**Navigation in der Objektstruktur, 1134**

Application, 1091, 1094

Dokument, 1117, 1123

MapForceView, 1132

Optionen, 1147

Projekt, 1151

**Navigieren,**

Lesezeichenbookmarks, 476

**negative, 794**

**Neue Zeile,**

in SQL WHERE-Anweisung, 434

Sonderzeichen - ersetzen, 253

**NewDocument, 1093, 1126**

**NewProject, 1093**

**Nicht druckbare Trennzeichen, 607**

**Nicht lateinischen Schriftsysteme, 1295**

**Nicht vorhanden,**

leere Felder als nicht vorhanden behandeln, 576

**Nilable,**

nicht unterstützt, 1030

**node, 566, 807**

Boolesche vergleichen, 260

Mappen fehlender Nodes, 283

mehrere summieren, 730

Position, 285

testen, 281

**node-name, 802**

**Nodes,**

duplizierte Nodes in Excel, 659

**Nodeset,**

Komplement, 32

**normalize-space, 776**

**normalize-unicode, 809**

**Not exist,**

Mappen fehlender Nodes, 283

**not-equal, 768**

**not-exists, 772**

**now, 790**

**Null,**

ersetzen, 428

Funktionen, 428

nillable, 332

xsi:nil, 332

**Null ersetzen,**

Datenbankfunktion, 428

**Null-Felder,**

leere Felder, 510

**number, 755, 807**

**numeric, 794**

## O

**Object tree navigation,**

AppOutputLine, 1097, 1101

AppOutputLines, 1102

AppOutputLineSymbol, 1104, 1105

Component, 1106, 1110

Components, 1112

Mapping, 1136, 1140

Mappings, 1141

**Objekt,**

Referenz, 1088

**Objektmodell,**

Übersicht, 1060

**Objektstrukturnavigation,**

Optionen, 1143

**Obligatorisch,**

Datenelemente für die XBRL-Instanz mappen, 868

**ODBC,**

native Verbindung, 387

**ODBC API,**

Verbindung herstellen über, 387

**Office Open XML,**

Excel 2007, 658

**Öffnen,**

MapForce Dateien in VS .NET, 879

**OnDocumentClosed, 1117**

**OnDocumentOpened, 1090**

**Online bereitstellen,**

Webservice, 825

**OnProjectClosed, 1149**

**OnProjectOpened, 1090**

**OOXML,**

Datenbank mappen auf, 681

Excel 2007, 658

mappen auf XML, 678

**Open,**

SQL script, 473

**OpenDocument, 1094, 1126**

**OpenProject, 1094**

**Optimale Größe,**

Doppelklick auf die Schaltfläche "Größe anpassen", 20

**Optimierung,**

Input-Verarbeitung aktivieren, 929

**Option,**

CSV-Dateioptionen, 510

**Optional,**

Eingabeparameter, 716

**Optionen, 1094, 1143**

allgemeine, 486

Ergebnisansicht - Datenbankabfrage, 489

für Codegenerierung, 1146

für die Codegenerierung, 1122, 1123

für Java, 1121

**Options,**

autocompletion - DB Query, 488

for code generation, 1146, 1148

text fonts - Database Query, 489

text fonts - DB Query, 487

**Oracle,**

Datenbankinfo, 409

Unterstützung, 59

**ORDER BY,**

SQL where-Komponente, 430

**Ordner,**

als globale Ressource, 167

Layout - Datenbankabfrage, 478

**Output,**

Datei - CSV, 510

Datei - Text, 522

mehrere Dateien, 190

XML-Instanz, 929

**Output encoding,**

default used, 1146

**Output-Symbol,**

Mapping, 96

**Override,**

context, 285

# P

**PADIS,**

IATA, 643

**Parameter,**

Ausgabe, 716

Befehlszeile, 266, 273

in Funktionen erweitern, 72, 136

in-context, 752

Input - dynamisch, 190

Input-Funktion, 273, 275

mittels Wildcards, 273

optionale, 716

SQL WHERE, 434

Standardwert, 275

und Codegenerierung, 273

**Parent, 1094, 1101, 1102, 1105, 1110, 1112, 1126, 1128, 1129, 1140, 1141, 1155**

Mappen und Filter, 234

**Parent-Kontext,**

Variable, 216

**parse-dateTime, 755****parse-number, 755****Parser,**

in Altova-Produkte integrierter, 1293

**Partner-Software, 963****Passwort,**

digitale Signatur, 225

**Path, 1123, 1155****PDF,**

Mapping-Dokumentation, 320

Stylevision Power Stylesheets, 314

**Pfad,**

absoluter bei Generierung von Code, 37

**Pfade,**

im generierten Code absolut machen, 205

**pi, 794****Plattformen für Altova-Produkte, 1292****Platzhalterzeichen,**

SQL WHERE, 434

**Plug-in,**

MapForce Editor, Ansicht, 900

MapForce für Eclipse, 884

MapForce für VS .NET, 878

MapForce Nature anwenden, 910

neues Mapping / Projekt erstellen, 904

Ordner examples importieren, 902

**position, 772, 804, 812**

node / Kontext, 285

**position() function,**

in XPath 1.0, 1246

**position() function in XPath 2.0,**

siehe fn:position(), 1254

**positive, 794****pow, 794****Pretty Print,**

in Ausgabekomponente, 929

**Priorität,**

Funktion, 72, 136

und Filter, 234

**Prioritätskontext,**

definieren, 262

**Programmiersprache,**

Enumerationen für, 1177

**Project type,**

for Java, 1147

**Projekt, 1149**

beim Öffnen, 1090

Dateiname, 1154

Dateiname und -pfad, 1152

neues erstellen, 1093

öffnen, 1094

Pfad mit Dateiname, 1155

speichern, 1156

Webservice erstellen, 822

**Projekttyp,**

Enumerationen für C#, 1179

für C#, 1145

**Protokolle,**

WSDL-unterstützte, 819

**Prozessor,**

Mapforce, 139, 156

**Prozessoren,**

zum Downloaden, 963

# Q

**Qname, 790**

String (Dimensions), 865

**QName Serialisierung,**

Rückgabe durch XPath 2.0 -Funktionen, 1255

**QName-as-string, 790**

**qname-related,**

Funktionen, 809

**Quelldatei,**

in mehrere Zieldateien aufteilen, 195

**Quelle,**

leere Felder als fehlend, 510

mehrere Quellen und Codegenerierung, 981

**Quellorientiertes Mapping,**

Mixed Content Mapping, 101

**Quellorientiertes Mapping im Vgl. zu  
Standard-Mapping, 107**

**Quit, 1095**

## R

**random, 794**

**Rangfolge,**

Trennzeichen in EDI-Dateien, 607

**Raster,**

Komponenten ausrichten, 78

**Rechtliches, 1298**

**Reduzieren,**

Bereich erweitern, 531

Regionen, 476

**Referenz, 916**

Funktionen in MapForce, 750

**Regex, 784**

**Regionen,**

einfügen, 476

entfernen, 476

erstellen, 476

erweitern, 476

reduzieren, 476

**Register,**

Stylevision-Ausgaberegister, 92

**Registrierung der Altova-Software, 961**

**Regular expressions, 784**

**Reihenfolge,**

der Verarbeitung von Komponenten, 152

**Rekursiv,**

Aufrufe in Funktionen, 697

benutzerdefinierte Funktion, 306

benutzerdefiniertes Mapping, 304

**Relativ,**

Pfade - Vorteile, 205

**remove-fileext, 764**

**remove-folder, 764**

**remove-timezone, 790**

**replace, 796, 809**

**replace-fileext, 764**

**Reservieren,**

Methodenname, 1030

**resolve-filepath, 764**

**resolve-uri, 802**

**Ressource,**

Datenbank, 178

Eigenschaften einer globalen Ressource, 183

Ordner, 167

**Rest einfügen,**

Tabellenaktion, 367

**reversefind-substring, 796**

**right, 796**

**right-trim, 796**

**Root,**

Element des Ziels, 259

Objekt - DB-Schema, 462

**Root-Element,**

neues erstellen, 114

**Root-Objekt,**

auswählen, 462

**Root-Tabellen, 357**

**round, 770**

**round-half-to-even, 808**

**round-precision, 770**

**Rows,**

mappen auf Textdateien, 503

mappen von - Textdateien, 506

**RTF,**

Mapping-Dokumentation, 320

Register im Mapping-Fenster, 68

Stylevision Power Stylesheets, 314

**Rückgängig,**

Änderungen an DB, 469

**Rückwärtskompatibilität,**

des XSLT 2.0-Prozessors, 1248

## S

**SAP,**

IDocs, 641

**Save, 1123, 1156**



**Save, 1123, 1156**

SQL scripts, 473

**SaveAs, 1124****Saved, 1124, 1156****Schattiert,**

Hintergrund, 1032

**Schema,**

Anzeigen mehrerer Ziele, 42

automatisch von XML-Datei generieren, 20

Code Generator, 966

Datenbank als Quelle, 59

Datenbank zuweisen, 438

in DB2-Komponente einbetten, 442

in IBM DB2 registriertes, 442

in SQL Server registriert, 455

in SQL-Server-Komponente zuweisen, 455

Katalogdateien, 298

mehrere Quellen, 929

mehrere Ziele, 37

Name, aus generiertem Code löschen, 929

Pfad zur Ausgabe hinzufügen, 929

rekursive Elemente, 304

Root-Objekt, 462

XBRL-Taxonomie einfügen, 847

**Schemafähigkeit,**

des XPath 2.0- und XQuery-Prozessors, 1254

**Schemanamen,**

aus Tabellennamen entfernen, 929

**schemanativetype, 1036****Schema-Validierung eines XML-Dokuments,**

für XQuery, 1251

**Schleifen,**

durchiterieren, 503

Gruppen und Hierarchien, 145

**Schlüssel,**

Felder in Textdateien, 506

Sortierschlüssel, 236

**Schlüsseleinstellungen,**

Tabellenaktionen, 398

**Schnellverbindung,**

Verbindungsassistent, 387

**Schriftart, 1295****Schriftsystem, 1295****Schriftsysteme, 1296****Script,**

ANT, 973

**Scripts in XSLT/XQuery,**

siehe Erweiterungsfunktionen, 1259

**second-from-datetime, 790****second-from-duration, 790****Sequenz,**

Position, 285

**set-empty, 775****Set-null,**

Datenbank-Nullfunktionen, 428

**Settings,**

autocompletion - DB Query, 488

text fonts - Database Query, 489

text fonts - DB Query, 487

**set-xsi-nil, 772****Shortcut,**

Tastatur, 950

**ShowItemTypes, 1134****ShowLibraryFunctionHeader, 1134****Signatur,**

Einstellungen, 225

**Signieren,**

digitale Signatur, 220

**SimpleType,**

sortieren, 236

**sin, 794****Soap,**

Webservice Fehler, 833

**Software-Produktlizenz, 1302****Solution-Datei, 975****Sonderzeichen,**

ersetzen, 253

**Sortieren,**

Daten im Ergebnisfenster, 483

Sortierkomponente, 236

Spaltensymbol im Ergebnisfenster, 483

Tabellen in Datenbankabfrage, 478

**Sortierreihenfolge,**

ändern, 236

**Sortierschlüssel,**

Sortierkomponente, 236

**Spalte,**

Überschriften - Erste Zeile als, 659

**Speichern,**

als Wert, 579

Daten im Ausgabefenster, 42

XML-Ausgabe, 139, 156

**SPL, 1034**

Codeblöcke, 1035

Dateien verwenden, 1040

foreach, 1043

**SPL, 1034**

- globale Objekte, 1039
- Variablen, 1038

**Sprachen,**

- und dynamische Unterstützung mehrerer, 188

**SPS,**

- benutzerdefinierte Stylesheets, 329
- Stylevision Power Stylesheet, 314
- vordefinierte Stylesheets zur Dokumentation von Mappings, 326

**SQL,**

- Anweisung in Tabellenaktionen, 398
- Anweisungen ausführen, 472
- Anweisungen generieren, 471
- Ausgabe neu generieren, 359
- Befehle im Ausgabe-Fenster, 359
- Daten vor Tabellenaktion löschen, 398
- Datenbanken direkt abfragen, 460
- Fenster in Datenbankabfrage, 470
- open script, 473
- Pseudo-SQL im Fenster Ausgabe-Vorschau, 359
- saving SQL scripts, 473

**SQL Editor,**

- Autokomplettierung, 474
- Einstellungen - allgemeine, 486
- Funktionen, 473
- Kommentare einfügen, 475
- Kommentare entfernen, 475
- Lesezeichen einfügen, 476
- Lesezeichen entfernen, 476
- Lesezeichen verwenden, 476
- Lesezeichenleiste, 476
- Regionen einfügen, 476
- Regionen entfernen, 476
- Regionen erstellen, 476
- Regionen verwenden, 476
- SQL ausführen, 472
- Text auskommentieren, 475

**SQL Script ausführen,**

- vom Register "Ausgabe"- , 438

**SQL Server,**

- Vorschau auf XML-Inhalt, 455
- XML-Daten mappen, 455
- XML-Schema einbetten, 455

**SQL Where,**

- Datentypfeld automatisch ermitteln, 434
- Komponente - einfügen, 430
- Operatoren, 434

ORDER BY, 430

Parameter, 434

XML-Daten abfragen, 448

Zeilenumbruch, 434

**SQL/XML,**

XML-Daten abfragen, 448

**SQL-Skript erneut ausführen,**

Ausgabe neu generieren, 359

**sqrt, 794****Standard,**

- benutzerdefinierte Funktion, 699
- Konfiguration - Globale Ressource, 160
- mapping method, 100
- Mixed Content Mapping, 107
- Parameter - Input-Komponente, 275

**Standardbibliothek,**

XSLT-Bibliothek, 72, 136

**Standardeinheiten,**

XBRL anzeigen, 868

**Standard-Mapping im Gegensatz zu quellorientiertem Mapping, 107****Standardwerte,**

- Dimension - identifier/period, 859
- XBRL, 851

**Start,**

der Applikation, 1095

**Starten,**

Plug-in für Eclipse, 897

**starts-with, 776, 809****Status, 1095, 1166****sting-compare-ignore-case, 796****Stream,**

von/auf Datenstreams mappen, 991

**Streaming,**

XML, CSV und FLF, 139

**String, 755, 802**

qname (Dimensions), 865

**string-compare, 796****string-join, 752****string-length, 776****Stylesheet,**

Stylevision Power Stylesheet Register, 92

**Stylesheets,**

für die Dokumentation definieren, 329

**StyleVision,**

- Befehlszeilenparameter, 266
- einer Komponente ein SPS zuweisen, 316
- HTML, PDF, RTF, Word2007+, 314

**StyleVision,**

- Mapping-Ausgaberegister, 92
- Register im Mapping-Fenster, 68
- SPS Stylesheets für Mappings definieren, 329
- Stylesheets in der Vorschau, 314
- verkettetes Mapping - Endkomponente, 124, 129
- XBRL-Taxonomie-Stylesheets, 846

**Sub-Einträge,**

- automatisch verbinden, 96

**substitute-missing, 772****substitute-missing-with-xsi-nil, 772****substring, 776****substring-after, 776, 809****substring-before, 776, 809****Subtract, 770, 805****Suchen,**

- Funktion in Bibliothek, 72, 136
- XSLT - Register Ausgabevorschau, 922

**sum, 752****Summieren,**

- Nodes in XSLT 1.0, 730

**Support für <MapForce, 963****Sybase,**

- Datenbankinfo, 413

**Symbole,**

- in Ergebnisfenster in Datenbankabfrage, 483
- in Meldungsfenster von Datenbankabfrage, 483

**Symboleiste,**

- globale Ressource, 159

**system-property, 814**

# T

**TA1-Segment, 602****Tabelle,**

- Aktionen - Datenbank, 400
- Beziehungen beibehalten/verwerfen, 417
- Daten in untergeordneten Tabellen ignorieren, 400
- gelöschte Daten vor Tabellenaktion, 398
- Hierarchie, 357
- Lookup - Wertezuordnung, 244
- Parent/Child anzeigen, 357
- Schemanamen entfernen, 929
- Tabellenaktionen - mehrere, 367
- Vorschau, 442, 455
- zu Komponente hinzufügen/ aus Komp. entfernen, 350

**Tabellen,**

- Symbol "Tabellenaktionen", 367

**Tabellen hinzufügen/entfernen,**

- in Datenbankkomponente, 350

**Tabellenaktion,**

- Rest einfügen, 367

**Tabellenaktionen,**

- Felder vergleichen, 400

**Tabellenbeziehungen, 357****Tabellendaten,**

- sortieren, 236

**Table,**

- context menu, 481

**tan, 794****Target-driven mapping, 100****Tastatur,**

- Kürzel, 950

**Tastaturkürzel,**

- Ausgabevorschau Zoomfaktor, 156
- Zoom-Faktor Fenster "Ausgabe", 139

**Tastenkürzel,**

- Shortcuts, 950

**Taxonomie,**

- XBRL-Taxonomie einfügen, 847

**Technische Informationen, 1291****Technischer Support für MapForce, 963****Terminologie, 12****Template,**

- aufrufen, 724
- named - summieren, 730

**Terminologie,**

- EDIFACT, 584

**Test,**

- Node-Test, 281

**Text,**

- auf Zeilen mappen, 503
- Dateien - Schlüsselfelder definieren, 506
- Iterator auf Zeilen mappen, 503
- mappen, 514
- neue Textdatei erstellen, 522
- Textdateien mappen, 498

**Textbegrenzung, 510, 522****Textinstanz,**

- FlexText Komponente, 548

**time-from-datetime, 790****Timeout,**

- iSeries - muss deaktiviert werden, 442

**time-to-xml, 801**

**timezone**, 790  
**Tipps**, 230  
**to-date**, 788  
**to-datetime**, 788  
**to-duration**, 788  
**Tokenize**, 776, 780  
**Tokenize-by-length**, 776, 780  
**tokenize-regexp**, 776  
**to-time**, 788  
**Transaktion**,  
    definieren / festlegen, 400  
**Transaktionen**,  
    X12 anpassen, 627  
**Transformationsergebnis**,  
    globale Ressourcen, 171  
**Transformieren**,  
    DoTransform.bat, 42  
    Input-Daten - Wertezuordnung, 244  
**translate**, 776  
**Treiber**,  
    JDBC, 929  
    Verbindungsassistent, 462  
**Trennen**,  
    einmal, 537  
**Trennzeichen**,  
    als vorhanden annehmen, 522  
    benutzerdefinierte, 607  
    EDI, 584  
    leere Felder als fehlend, 510  
    nicht druckbare, 607  
    Rangfolge in EDI-Dateien, 607  
    Unterkomponenten-Trennzeichen, 607  
**Trennzeichengetrennte Dateien**,  
    CSV, 499  
**true**, 803  
**Tutorial**, 18  
    FlexText, 534  
    Ordner examples, 18  
**TXT**,  
    Feld-Datentypen, 510  
**Typ**,  
    in Zieltyp konvertieren, 929  
**Typen**,  
    derived types - xsi:type, 302  
    integrierte, 1049  
**Typkonvertierung**,  
    überprüfen, 295  
**Typorientiert**,

Verbindungen, 109

## U

**Über MapForce**, 963  
**Überprüfen**,  
    Typ überprüfen, 295  
**Überschriften**,  
    Excel-Zeile als Spaltenüberschriften, 659  
**Übersicht**, 531  
    über MapForce API, 1059  
**Übersicht über MapForce**, 10  
**UCS-2**, 1295  
**UCS-4**, 1295  
**Uhrzeit**,  
    xsd:time, 607  
**UI**,  
    Komponente definieren, 737  
**UN/EDIFACT**, 582  
    als Ziel, 607  
    angepasste Datei, 625  
    anpassen, 617  
    anpassen - Konfiguration, 625  
    anpassen - vorbereiten, 618  
    auf XML mappen, 585  
    Autokomplettierungsregeln, 611  
    Terminologie, 584  
    Validierungsregeln - Ziel, 611  
**unary-minus**, 794  
**Unicode**,  
    Code Point Collation, 236  
    Konsortium, 1295  
    Standard, 1295  
    Unterstützung, 1295  
**Unicode-Zeichen**, 1295  
**Universalschriftart**, 1295  
**unparsed-entity-uri**, 814  
**Untergeordnete**,  
    Verbindungen - inkludieren, 114  
**Untergeordnete Daten**,  
    in untergeordneten Tabellen ignorieren, 400  
**Untergeordnetes Feld**,  
    Trennzeichen, 607  
**Unterstützung**,  
    Datenbanken, 338  
    Datenbankinfo, 404

**uppercase, 796**

**upper-case, 809**

**URI,**

in DTDs, 294

**UTF-8, 1295**

## V

**Validator,**

in Altova-Produkten, 1293

**Validieren,**

Daten im Ausgabefenster, 42

Mapping-Projekt, 142

Output-Daten, 142

XML, 139, 156

**Validierung,**

von ausgeführtem Code, 607

**Variable,**

SQL WHERE Parameter, 434

**Variablen,**

in SPL, 1038

**Verarbeitungsreihenfolge,**

von Komponenten in einem Mapping, 152

**Verbinden,**

Datenbankabfrage - Verbindung herstellen, 462

über die ODBC API, 387

**Verbindung, 938**

Assistent, 387

Eigenschaften, 114

JDBC, 394

löschen, 114

nativ, 387

Parent/Child-Konnektoren verschieben, 114

**Verbindungen,**

typorientiert, 109

**Verbindungsassistent, 462**

Verbindungen herstellen, 387

**Vergrößern/Verkleinern,**

Faktor im Fenster Ausgabevorschau, 156

**Verketteten,**

Filter - vermeiden, 234

**Verkettetes,**

Mapping - Codegenerierung, 129

**Verkettetes Mapping,**

Endkomponente mittels StyleVision anzeigen, 124, 129

**Verschachtelt,**

benutzerdefinierte Funktionen, 716

**Verschieben,**

Dimension in XBRL-Komponente, 859

Parent/Child-Konnektoren, 114

**Version, 1092, 1093**

Kompatibilität der Wrapper Klassen, 950

**Vertrieb,**

von Altova Produken, 1301

von Altova Software-Produkten, 1298

von Altova-Software-Produkten, 1299

**Verwenden,**

Globale Ressourcen, 165

**Verwendete Funktionen, 326**

**Visible, 1095**

**Visual Basic,**

Fehlerbehandlung, 1071

Integration von MapForce, 1206

**Visual Studio,**

unterstützte Versionen - Codegenerierung, 950

**Visual Studio .Net,**

MapForce Plug-in, 878

MapForce-Dateien öffnen, 879

und XMLSPY - Unterschiede, 881

**Visueller Function Builder, 687**

**Von rechts nach links, 1296**

**Voraussetzungen,**

Webservices, 818

**Vorlage,**

FlexText in MapForce verwenden, 545

**Vorschau,**

Mapforce-Prozessor, 139, 156

Stylevision Power Stylesheets, 314

Tabellen und Inhalt, 442, 455

Wert der Input-Komponente, 275

**VS .NET,**

C# Webservice erstellen/online bereitstellen, 831

C# Webservice kompilieren, 831

**VS NET 2002/2003, 975**

## W

**Warnung,**

Validierung, 142

**Webservice,**

aufrufen, 836

C# erstellen, 831

**Webservice,**

- Fehler, 833
- Java erstellen, 825
- Java online bereitstellen, 825
- Projekt erstellen, 822

**Webservice-Aufruf,**

- einfügen, 837, 842

**Webservices,**

- Voraussetzungen, 818

**Wechseln,**

- Konfiguration - globale Ressource, 165

**weekday, 790****weeknumber, 790****Wert,**

- default, 716
- Input-Parameter - Vorschau, 275

**Wert definieren,**

- Input-Komponente - Vorschau, 275

**Wert speichern, 579****Wertezuordnung,**

- Daten unverändert übergeben, 248
- Lookup-Tabelle, 244
- Lookup-Tabelle - Eigenschaften, 251
- und Hypercube Dimensions, 865
- XBRL, 865

**Where,**

- SQL WHERE Operator, 434
- SQL WHERE-Komponente, 430
- XML-Daten in DB2 abfragen, 448

**Whitespace im XML-Dokument,**

- Behandlung durch den Altova XSLT 2.0-Prozessor, 1248

**Whitespace Nodes im XML-Dokument,**

- und Behandlung durch den XSLT 1.0-Prozessor, 1246

**Whitespace-Behandlung,**

- und XPath 2.0-Funktionen, 1254

**Wiedergabe, 1296****Wildcards,**

- Verwendung von Anführungszeichen in der Befehlszeile, 273

**WindowHandle, 1096****Windows,**

- Unterstützung für Altova-Produkte, 1292

**Windows 2000, 1295****Windows NT, 1295****Wirkungsanalyse,**

- SPS Stylesheet, 326

**Word,**

- Mapping-Dokumentation, 320

**Word2007+,**

- Register im Mapping-Fenster, 68
- Stylevision Power Stylesheets, 314

**Wrapper,**

- Klassen, 950

**Wrapper-Klassen,**

- Versionskompatibilität, 950

**WSDL,**

- 2.0-Unterstützung, 819
- Input/Output Webservice-Aufruf, 837, 842
- unterstützte Protokolle, 819

## X

**X12,**

- 997 Functional Acknowledgement, 598
- 997 Funktionsbestätigung, 598
- 999 Implementation Acknowledgement, 600
- als Ziel, 607
- Autokomplettierungsregeln, 614
- Codegenerierung C++-Klasse, 625
- config-Datei für mehrere Nachrichten aktualisieren, 639
- globale Anpassung, 632
- HIPAA, 646
- Validierungsregeln - Ziel, 614
- Vorbereitung für die Anpassung, 630
- zusammengeführte Einträge, 628
- zusammengeführte Einträge aufgliedern, 636

**X12 Acknowledgment,**

- TA1 Segment, 602

**XBRL, 800**

- Datenbank auf XBRL-Taxonomie, 871
- Dokument / Taxonomie einfügen, 847
- Excel auf XBRL - Beispiel, 867
- Hypercubes - Domains/Dimensions, 859
- Input/Output-Dateien mappen, 846
- Komponentenstruktur / Anzeige, 851
- obligatorische Datenelemente für die Instanzdatei, 868
- Standardwerte (items), 851
- Wertezuordnung, 865

**XBRL Taxonomie-Stylesheets in StyleVision, 846****xbml-measure-currency, 800****xbml-measure-pure, 800****xbml-measures-shares, 800****Xerces,**

- Bibliotheken, 950

**Xerces,**

Unterstützung, 1032

**xlsx,**

Bibliotheksfunktionen, 801

OOXML, 658

**XML, 1295**

auf IBM DB2-Zielkomponente mappen, 450

Ausgabe speichern, 139, 156

Ausgabe validieren, 139, 156

Datenbank mappen auf OOXML, 681

in DB2 abfragen, 448

mehrere Tabellen mappen, 257

OOXML mappen auf, 678

Signatureinstellungen, 225

Streaming, 139

von UN/EDIFACT mappen, 585

Vorschau auf XML-Inhalt, 455

XML-Daten mappen und abfragen, 448

XML-Daten von DB2 mappen, 442

XML-Inhalt, 442

XML-Schema von XML-Datei generieren, 20

**XML auf Datenbank,**

mappen, 350

**XML Parser,**

Info, 1293

**XML-Dateien,**

anhand einer einzigen XML-Quelle, 195

von Datenbank generieren, 198

von Excel generieren, 200

**xmlexists,**

Abfragefunktion, 448

**XML-Instanz,**

absoluter Pfad, 37

Input, 929

Output, 929

**XML-Schema,**

automatisch generieren, 20

einer Datenbank zuweisen, 438

in DB2-Komponente einbetten, 442

in IBM DB2 registriertes, 442

in SQL Server registriert, 455

in SQL-Server-Komponente zuweisen, 455

**xpath, 812**

mehrere Nodes summieren, 730

XML-Abfrage in DB2, 448

**XPath 2.0 Functions,**

implementation information, 1254

**XPath 2.0-Funktionen,**

allgemeine Informationen, 1254

spezifische Funktionen siehe unter fn:, 1254

**xpath2,**

Bibliothek, 72, 136

Bibliotheksfunktionen, 802

**XPath-Funktionen - Unterstützung,**

für einzelne Funktionen siehe unter fn:, 1255

**XQuery,**

benutzerdefinierte Funktionen hinzufügen, 729

Erweiterungsfunktionen, 1259

Funktionen, 72, 136

und Datenbanken, 338

**XQuery 1.0 Functions,**

implementation information, 1254

**XQuery 1.0-Funktionen,**

allgemeine Informationen, 1254

spezifische Funktionen siehe unter fn:, 1254

**XQuery 1.0-Prozessor,**

Informationen, 1251

**XQuery-Prozessor,**

in Altova-Produkten, 1294

**xs:,**

constructors, 803

Konstrukturen, 803

**xs:date, 804****xs:QName,**

siehe auch QName, 1255

**xs:time, 804****xsd,**

Datum/Uhrzeit, 607

**Xsi:nil,**

nillable, 332

**xsi:type,**

mappen auf derived types, 302

**xsl:preserve-space, 1246****xsl:strip-space, 1246****XSLT,**

benutzerdefinierte Funktionen hinzufügen, 724

Bibliotheksfunktionen, 812

Codegenerierung, 1120

Erweiterungsfunktionen, 1259

generieren (Tutorial), 36

mehrere Ziele generieren, 42

Optionen, 1148

Register im Mapping-Fenster, 68

Standardbibliothek, 72, 136

Template Namespace, 724

**XSLT 1.0-Prozessor,**

**XSLT 1.0-Prozessor,**

Einschränkungen und implementierungsspezifisches Verhalten, 1246

**XSLT 2.0,**

benutzerdefinierte Funktionen hinzufügen, 728

**XSLT 2.0 Stylesheet,**

Namespace-Deklarationen, 1248

**XSLT 2.0-Funktionen,**

implementierungsspezifisches Verhalten, 1250  
spezifische Funktionen siehe unter fn:, 1250

**XSLT 2.0-Prozessor,**

allgemeine Informationen, 1248  
Informationen, 1248

**XSLT2.0,**

Date-Konstruktor, 728

**XSLT-Prozessoren,**

in Altova-Produkten, 1294

**xlsx-to-date, 801****xlsx-to-datetime, 801****xlsx-to-time, 801**

## Y

**year-from-datetime, 790****year-from-duration, 790**

## Z

**Z bis A,**

Sortierkomponente, 236

**Zeichen,**

füllen, 514

**Zeichen-Entities,**

in der HTML-Ausgabe von XSLT-Transformationen, 1246

**Zeichennormalisierung,**

in XQuery-Dokument, 1251

**Zeile,**

als Spaltenüberschrift, 659

**Zeilen,**

mappen auf Textdateien, 503  
mappen von - Textdateien, 506  
von Excel, 200

**Zeilenbasiert,**

einmal trennen, 560  
mehrmals trennen, 555

**Zeilenkommentar, 475****Zeilenumbruch,**

in SQL WHERE-Anweisung, 434

**Zeilenumbrüche,**

Sonderzeichen ersetzen, 253

**Zertifikat,**

digitale Signaturen, 225

**Ziel,**

EDI-Dokumente, 607  
Komponente IBM DB2, 450  
mehrere Schemas, 37  
mehrere Schemas anzeigen, 42  
mehrere Ziele und Codegenerierung, 981  
Root-Element, 259

**Zieldatei,**

mehrere Zieldateien anhand einer einzigen Quelldatei, 195

**Zielelement,**

mehrere Quellen mappen, 44

**Zielkomponente,**

FlexText Komponente, 549

**Zielorientiert,**

Mapping, 107

**Zoom,**

Faktor im Ausgabefenster, 139

**Zu wenig Arbeitsspeicher, 1030****Zusammenführen,**

mehrere Input-Dateien, 190  
XML -Dateien, 264

**Zusammengeführte Einträge, 628**

in separate Datenelemente aufgliedern - X12, 636

**Zuweisen,**

Globale Ressource zu einer Komponente, 163  
Stylevision Power Stylesheet zu einer Komponente, 316