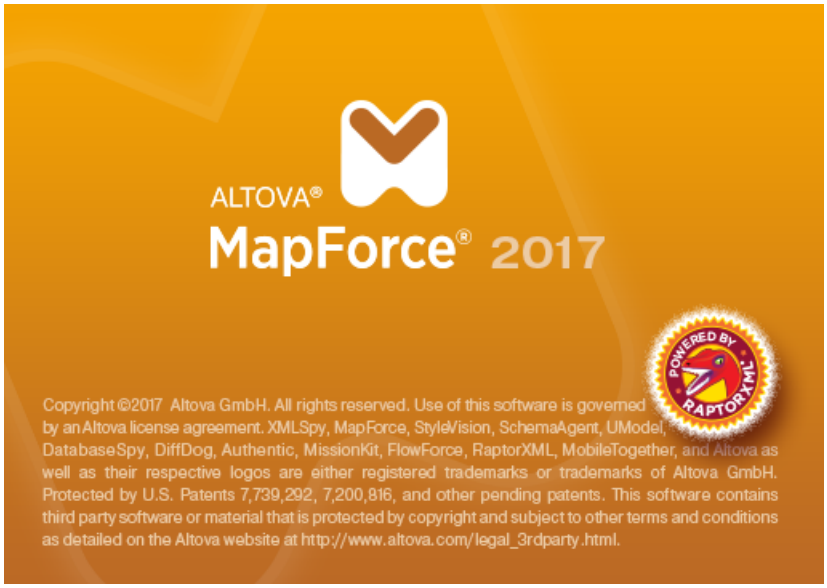
ユーザーマニュアル


ALTOVA®
MapForce® 2017
POWERED BY RAPTORXML
Copyright ©2017 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, MobileTogether, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third 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 2017 Enterprise Edition
ユーザーマニュアル

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.

発行日 : 2017

(C) 2017 Altova GmbH

目次

1 Altova MapForce 2017 Enterprise Edition 3

1.1 新規機能 4

2 はじめに 12

2.1 MapForce とは ? 13
2.2 基本概念 18
2.3 ユーザーインターフェイスの概要 20
2.4 規約 28

3 チュートリアル 30

3.1 XML を新しいスキーマに変換 31
3.2 複数のソースから1つのターゲットにマップ 40
3.3 複数のターゲットスキーマとの作業 45
3.4 ファイルを動的に処理と生成する 52

4 一般的なタスク 62

4.1 マッピングとの作業 63
4.1.1 マッピングにコンポーネントを追加する 63
4.1.2 URL からコンポーネントを追加する 64
4.1.3 データストリーミングについて 66
4.1.4 変換言語の選択 67
4.1.5 マッピングの検証 68
4.1.6 マッピング出力の検証 69
4.1.7 出力のプレビュー 70
4.1.8 テキストビューの機能 71
4.1.9 テキストビュー内の検索 75
4.1.10 XSLT コードのプレビュー 79
4.1.11 XSLT コードの生成 79
4.1.12 XQuery コードのプレビュー 80
4.1.13 複数のマッピングウィンドウとの作業 80

4.1.14 マッピング設定の変更 ... 81
4.2 コンポーネントとの作業 ... 84
4.2.1 コンポーネント内の検索 ... 85
4.2.2 コンポーネントの整列 ... 86
4.2.3 コンポーネント設定を変更する ... 87
4.2.4 入力の複製 ... 87
4.3 接続との作業 ... 89
4.3.1 必須の入力について ... 90
4.3.2 優先する接続線の表示の変更 ... 91
4.3.3 接続の注釈 ... 92
4.3.4 接続設定 ... 92
4.3.5 マッチする子要素の接続 ... 94
4.3.6 コンテキストメニューの接続 ... 96
4.3.7 見つからない親接続の通知 ... 97
4.3.8 接続と子接続の移動 ... 98
4.3.9 コンポーネント削除後の接続の保持 ... 101
4.3.10 見つからないアイテムの扱い方 ... 102
4.4 マッピングプロジェクトとの作業 ... 106
4.4.1 プロジェクトを開く、検索する、閉じる ... 107
4.4.2 新規プロジェクトの作成 ... 107
4.4.3 コード生成設定の設定 ... 108
4.4.4 プロジェクトフォルダーの管理 ... 109

5 マッピングのデザイン 112

5.1 相対と絶対パスの使用 ... 113
5.1.1 コンポーネント上で相対パスを使用する ... 113
5.1.2 ファイルベースのデータベースへパスを設定する ... 115
5.1.3 破損したパスの参照を修正する ... 117
5.1.4 生成されたコード内のパスについて ... 117
5.1.5 コピーと貼り付けと相対パス ... 118
5.2 マッピング接続の種類 ... 119
5.2.1 ターゲット優先マッピング ... 119
5.2.2 ソース優先マッピング ... 119
5.2.3 全てコピー接続 ... 126
5.3 チェーンマッピング ... 128
5.3.1 サンプル：パススルーが有効な場合 ... 129

5.3.2 サンプル :パススルーが無効な場合 ... 133
5.4 複数の入力または出力ファイルを動的に処理 ... 136
5.4.1 複数の入力ファイルを単一の出力ファイルにマップする ... 138
5.4.2 複数の入力ファイルを複数の出力ファイルにマッピングする ... 139
5.4.3 ファイル名をマッピングパラメーターとして提供する ... 140
5.4.4 複数の出力ファイルをプレビューする ... 140
5.4.5 サンプル :1つの XML ファイルを複数のファイルに分割 ... 141
5.4.6 サンプル :データベーステーブルに対して複数の XML を生成 ... 143
5.4.7 Excel の行から複数の XML ファイルを生成 ... 144
5.5 マッピングにパラメーターを与える ... 150
5.5.1 単純型入力コンポーネントの追加 ... 150
5.5.2 単純型入力コンポーネント設定 ... 151
5.5.3 デフォルトの入力値を作成する ... 152
5.5.4 例 :名前をマッピングパラメーターとして使用する ... 153
5.6 マッピングから文字列の値を返す ... 156
5.6.1 単純型出力コンポーネントの追加 ... 157
5.6.2 サンプル :関数出力のプレビュー ... 158
5.7 変数の使用 ... 160
5.7.1 変数の追加 ... 161
5.7.2 変数のコンテキストとスコープの変更 ... 164
5.7.3 サンプル :データベースのテーブルの行の計算 ... 166
5.7.4 サンプル :データベースのテーブルの行の計算 ... 167
5.7.5 サンプル 記録のグループ化、および、サブグループ化 ... 168
5.8 データの並べ替え ... 170
5.8.1 複数のキーを使用した並べ替え ... 172
5.8.2 変数を使用して並べ替える ... 173
5.9 フィルターと条件 ... 175
5.9.1 サンプル :ノードのフィルター ... 176
5.9.2 サンプル :条件付で値を返す ... 178
5.10 データのジョイン ... 181
5.10.1 ジョインの条件の追加 ... 183
5.10.2 3つ以上の構造をジョインする ... 185
5.10.3 サンプル : XML 構造のジョイン ... 186
5.11 Value-Map の使用 ... 191
5.11.1 値を変更せずに Value-Map を通過させる ... 194
5.11.2 Value-Map コンポーネントプロパティ ... 196
5.12 例外の追加 ... 199

5.12.1 例 :"より大きい "条件の例外 ... 200
5.12.2 例 :ノードが存在しない場合の例外 ... 200
5.13 文字列の解析とシリアル化 ... 203
5.13.1 コンポーネントの解析 /シリアル化について ... 203
5.13.2 サンプル :文字列の解析 (固定長テキストからExcel へ) ... 205
5.13.3 サンプル :文字列にシリアル化 (XML からデータベースへ) ... 210
5.14 ノード名のマッピング ... 215
5.14.1 ノード名へのアクセスを取得する ... 216
5.14.2 特定の型のノードへのアクセス ... 223
5.14.3 サンプル :要素名を属性値にマップする ... 226
5.14.4 サンプル :名前別にノードをグループ化またはフィルターする ... 230
5.14.5 マッピングのルールと戦略 ... 233

6 マッピングのデバッグ 248

6.1 デバッガーの準備 ... 251
6.2 デバッガーコマンド ... 252
6.2.1 デバッグモードについて ... 253
6.3 ブレークポイントの追加と削除 ... 256
6.4 値ウィンドウの使用 ... 258
6.5 コンテキストウィンドウの使用 ... 260
6.5.1 ブレークポイントウィンドウの使用 ... 261
6.6 部分的に生成された出力のプレビュー ... 263
6.7 コネクタの現在の値を表示する ... 264
6.8 最近の過去にステップバックする ... 265
6.9 コネクタにより処理された値の履歴を確認する ... 266
6.10 コンテキストを値に設定する ... 267
6.11 デバッガー設定 ... 268

7 データソースとターゲット 270

7.1 XML と XML スキーマ ... 271
7.1.1 XML スキーマの生成 ... 271
7.1.2 XML コンポーネント設定 ... 271
7.1.3 "スキーマ "コンポーネントとしてを DTD 使用する ... 275
7.1.4 派生した XML スキーマの型 ... 275
7.1.5 QNames ... 277
7.1.6 Nil の値 /Nillable ... 278

7.1.7 コメントと処理命令 ... 281
7.1.8 CDATA セクション ... 282
7.1.9 ワイルドカード - xs:any / xs:anyAttribute ... 283
7.1.10 複数のスキーマからのデータのマージ ... 287
7.1.11 カスタム名前空間の宣言 ... 288
7.1.12 デジタル署名 ... 290
7.2 MapForce とデータベース ... 298
7.2.1 データベースへの接続 ... 300
7.2.2 Linux と Mac 上でのデータベース接続 ... 355
7.2.3 データベースマッピングの概要 ... 357
7.2.4 データをデータベースにマップする ... 383
7.2.5 データベースのデータのジョイン ... 423
7.2.6 データベースデータのフィルターと並べ替え (SQL WHERE/ ORDER) ... 437
7.2.7 仮想テーブルの SQL SELECT ステートメント ... 443
7.2.8 XML データから / データベースフィールドへのマッピング ... 451
7.2.9 データベースの参照とクエリ ... 460
7.2.10 ストアドプロシージャ ... 476
7.3 CSV とテキストファイル ... 495
7.3.1 サンプル : CSV ファイルから XML へのマッピング ... 495
7.3.2 サンプル : アイテムの反復 ... 498
7.3.3 サンプル : CSV ならびに固定長テキストファイルから階層構造を作成 ... 499
7.3.4 CSV オプションの設定 ... 502
7.3.5 サンプル : 固定長テキストファイルを (データベースへ) マッピング ... 506
7.3.6 FLF オプションの設定 ... 513
7.4 MapForce FlexText ... 520
7.4.1 概要 ... 520
7.4.2 FlexText チュートリアル ... 522
7.4.3 FlexText コンポーネント設定 ... 532
7.4.4 ターゲットコンポーネントとして FlexText を使用する ... 534
7.4.5 FlexText レファレンス ... 535
7.4.6 FlexText と正規表現 ... 558
7.5 EDI ... 565
7.5.1 EDI 用語解説 ... 567
7.5.2 EDI コンポーネントをマッピングに追加する ... 567
7.5.3 EDI コンポーネント設定 ... 569
7.5.4 EDI コンポーネント検証 ... 572
7.5.5 ASC X12 ... 575

7.5.6 HIPAA X12 579
7.5.7 HL7 Version 2 584
7.5.8 IATA PADIS 586
7.5.9 SAP IDoc 590
7.5.10 TRADACOMS 592
7.5.11 UN/EDIFACT 606
7.5.12 EDI 構造のカスタム化 614
7.5.13 構成ファイルを使用して EDI 検証をカスタマイズする 642
7.5.14 JSON 650
7.5.15 MS OOXML Excel 2007 以降のファイル 661
7.5.16 XBRL 680
7.5.17 HL7 Version 3 714

8 関数 718

8.1 関数と作業 719
8.2 ユーザー定義関数 722
8.2.1 関数のパラメーター 727
8.2.2 インラインと一般的なユーザー定義関数 729
8.2.3 単純型のルックアップ関数を作成する 731
8.2.4 ユーザー定義関数 - サンプル 734
8.2.5 複合型のユーザー定義関数 - XML ノードを入力に使用 739
8.2.6 複合型のユーザー定義関数 - XML ノードを出力に使用 743
8.2.7 再帰的なユーザー定義マッピング 748
8.3 カスタム XSLT 1.0 または 2.0 関数のインポート 754
8.3.1 例 : カスタム XSLT 1.0 関数の追加 755
8.4 カスタム XQuery 1.0 関数をインポートする 758
8.5 カスタム Java と .NET ライブラリのインポート 759
8.5.1 例 : カスタム Java クラスのインポート 761
8.5.2 例 : カスタム .NET DLL アセンブリのインポート 762
8.6 Java、C# および C++ ライブラリを手動で参照する 763
8.6.1 .mff ファイルの構成 763
8.6.2 .mff ファイルを MapForce にインポートする 768
8.6.3 データ型のマッピング 768
8.6.4 例 : カスタム C# ライブラリの作成 770
8.6.5 例 : カスタム C++ ライブラリの作成 771
8.6.6 例 : カスタム Java ライブラリの作成 774

8.7 正規表現 ... 776
8.8 関数ライブラリレファレンス ... 779
8.8.1 core | QName functions ... 779
8.8.2 core | aggregate functions ... 781
8.8.3 core | conversion functions ... 785
8.8.4 core | file path functions ... 794
8.8.5 core | generator functions ... 796
8.8.6 core | logical functions ... 798
8.8.7 core | math functions ... 801
8.8.8 core | node functions ... 803
8.8.9 core | sequence functions ... 805
8.8.10 core | string functions ... 818
8.8.11 db ... 825
8.8.12 edifact ... 827
8.8.13 lang | QName functions ... 829
8.8.14 lang | datetime functions ... 829
8.8.15 lang | generator functions ... 842
8.8.16 lang | logical functions ... 842
8.8.17 lang | math functions ... 843
8.8.18 lang | string functions ... 846
8.8.19 xbrl ... 850
8.8.20 xlsx ... 851
8.8.21 xpath2 | accessors ... 853
8.8.22 xpath2 | anyURI functions ... 853
8.8.23 xpath2 | boolean functions ... 854
8.8.24 xpath2 | constructors ... 854
8.8.25 xpath2 | context functions ... 855
8.8.26 xpath2 | durations, date and time functions ... 856
8.8.27 xpath2 | node functions ... 857
8.8.28 xpath2 | numeric functions ... 859
8.8.29 xpath2 | qname-related functions ... 859
8.8.30 xpath2 | string functions ... 859
8.8.31 xslt | xpath functions ... 861
8.8.32 xslt | xslt functions ... 863

9 Web サービスの実装 868

9.1 WSDL サポート情報 ... 871
9.2 WSDL ファイルからWeb サービスプロジェクトを作成 ... 874
9.3 MapForce にて Java Web サービスを生成する ... 877
9.3.1 Web サービスを使用する - getPerson オペレーション ... 879
9.3.2 Web サービスを使用する - putPerson オペレーション ... 881
9.4 MapForce にて C# ウェブサービスを生成 ... 883
9.5 Web サービスのフォールト (fault) ... 885

10 Web サービスの呼び出し 888

10.1 Web サービス呼び出しの追加 (REST-style) ... 890
10.2 Web サービス呼び出しの追加 (WSDL-Style) ... 901
10.3 WSDL 呼び出し設定 ... 902
10.4 HTTP セキュリティの設定 ... 906
10.5 WS-Security の設定 ... 908
10.6 例 : REST-スタイル Web サービスの呼び出し ... 911
10.7 例 : RSS フィードからのデータのマッピング ... 914
10.8 サンプル : WSDL-スタイル Web サービスの呼び出し ... 918
10.9 Digital Certificate Management ... 921
10.9.1 Trusting Server Certificates on Linux ... 923
10.9.2 Trusting Server Certificates on Mac ... 926
10.9.3 Trusting Server Certificates on Windows ... 927
10.9.4 Accessing the Certificate Stores on Windows ... 929
10.9.5 Exporting Certificates from Windows ... 930
10.9.6 Client Certificates on Linux ... 933
10.9.7 Client Certificates on Mac ... 934
10.9.8 Client Certificates on Windows ... 935

11 MapForce とマッピングの自動化 938

11.1 MapForce Server について ... 939
11.2 MapForce Server 実行可能ファイルにマッピングをコンパイルする ... 940
11.3 マッピングを FlowForce Server にデプロイ ... 942
11.4 RaptorXML Server を使用して自動化する ... 946
11.5 MapForce コマンドラインインターフェイス ... 947

12 Map のカスタム化 954

12.1 MapForce オプションの変更 ... 955
12.2 Altova グローバルリソース ... 957
12.2.1 グローバルリソース - ファイル ... 957
12.2.2 グローバルリソース - フォルダー ... 962
12.2.3 グローバルリソース - アプリケーションワークフロー ... 964
12.2.4 グローバルリソース - データベース ... 970
12.2.5 グローバルリソース - プロパティ ... 973
12.3 StyleVision を使用してマッピングの出力のスタイルを指定する ... 977
12.3.1 StyleVision スタイルシートを使用したマッピングの例 ... 978
12.4 マッピングドキュメントの生成とカスタム化 ... 981
12.4.1 定義済みの SPS スタイルシート ... 985
12.4.2 カスタムデザイン ... 988
12.5 キーボードのショットカットのカスタム化 ... 990
12.6 カタログファイル ... 992

13 Visual Studio のための MapForce プラグイン 998
13.1 プラグインの有効化 ... 999
13.2 プラグインの有効化 ... 1001
13.3 共通メニューと関数へのアクセス ... 1003

14 Eclipse のための MapForce プラグイン 1006
14.1 Eclipse のための MapForce プラグインのインストール ... 1007
14.2 MapForce パースペクティブ ... 1011
14.3 共通メニューと関数へのアクセス ... 1014
14.4 マッピングおよびプロジェクトと作業 ... 1017
14.4.1 MapForce/ Eclipse プロジェクトの作成 ... 1017
14.4.2 新しいマッピングの作成 ... 1019
14.4.3 Eclipse プロジェクトに既存のマッピングをインポートする ... 1020
14.4.4 自動ビルドの構成と MapForce コードの生成 ... 1022
14.5 Eclipse のための MapForce プラグインの有効化 ... 1024

15 メニューレファレンス 1030
15.1 ファイル ... 1031
15.2 編集 ... 1033
15.3 挿入 ... 1034

15.4 プロジェクト ... 1036
15.5 コンポーネント ... 1038
15.6 接続 ... 1040
15.7 関数 ... 1041
15.8 出力 ... 1042
15.9 デバッグ ... 1043
15.10 表示 ... 1044
15.11 ツール ... 1046
15.12 ウィンドウ ... 1047
15.13 ヘルプメニュー ... 1048

16 Code Generator 1054

16.1 Introduction to code generator ... 1055
16.2 What's new 1057
16.3 Generating C++ code ... 1059
16.3.1 Generating code from a mapping ... 1060
16.3.2 Generating code from a mapping project ... 1060
16.3.3 Building the project ... 1061
16.3.4 Running the application ... 1061
16.4 Generating C# code ... 1063
16.4.1 Generating code from a mapping ... 1064
16.4.2 Generating code from a mapping project ... 1064
16.4.3 Building the project ... 1065
16.4.4 Running the application ... 1065
16.5 Generating Java code ... 1066
16.5.1 Generating code from a mapping ... 1067
16.5.2 Generating code from a mapping project ... 1067
16.5.3 Handling JDBC references ... 1068
16.5.4 Building the project with Ant ... 1069
16.5.5 Example: Build a Java application with Eclipse and Ant ... 1070
16.6 Integrating MapForce-Generated Code ... 1078
16.6.1 Java example ... 1079
16.6.2 C# example ... 1080
16.6.3 C++ example ... 1082
16.6.4 Changing the data type of the mapping input/output (C#, Java) ... 1083
16.7 Generating Code from XML Schemas or DTDs ... 1088

16.7.1 About Schema Wrapper Libraries (C++) ... 1090
16.7.2 About Schema Wrapper Libraries (C#) ... 1093
16.7.3 About Schema Wrapper Libraries (Java) ... 1095
16.7.4 Integrating Schema Wrapper Libraries ... 1097
16.7.5 Example: Using the Schema Wrapper Libraries ... 1100
16.8 Reference to Generated Classes (C++) ... 1124
16.8.1 altova::DateTime ... 1124
16.8.2 altova::Duration ... 1127
16.8.3 altova::DayTimeDuration ... 1129
16.8.4 altova::YearMonthDuration ... 1130
16.8.5 altova::meta::Attribute ... 1130
16.8.6 altova::meta::ComplexType ... 1131
16.8.7 altova::meta::Element ... 1132
16.8.8 altova::meta::SimpleType ... 1133
16.8.9 [YourSchema]::[CDoc] ... 1134
16.8.10 [YourSchema]::MemberAttribute ... 1136
16.8.11 [YourSchema]::MemberElement ... 1137
16.9 Reference to Generated Classes (C#) ... 1138
16.9.1 Altova.Types.DateTime ... 1138
16.9.2 Altova.Types.DateTimeFormat ... 1141
16.9.3 Altova.Types.Duration ... 1142
16.9.4 Altova.Xml.Meta.Attribute ... 1145
16.9.5 Altova.Xml.Meta.ComplexType ... 1145
16.9.6 Altova.Xml.Meta.Element ... 1146
16.9.7 Altova.Xml.Meta.SimpleType ... 1146
16.9.8 [YourSchema].[Doc] ... 1147
16.9.9 [YourSchemaType].MemberAttribute ... 1149
16.9.10 [YourSchemaType].MemberElement ... 1150
16.10 Reference to Generated Classes (Java) ... 1152
16.10.1 com.altova.types.DateTime ... 1152
16.10.2 com.altova.types.Duration ... 1156
16.10.3 com.altova.xml.meta.Attribute ... 1160
16.10.4 com.altova.xml.meta.ComplexType ... 1160
16.10.5 com.altova.xml.meta.Element ... 1161
16.10.6 com.altova.xml.meta.SimpleType ... 1161
16.10.7 com.[YourSchema].[Doc] ... 1162
16.10.8 com.[YourSchema].[YourSchemaType].MemberAttribute ... 1164

16.10.9 com.[YourSchema].[YourSchemaType].MemberElement 1164
16.11 Code Generation Tips 1166
16.12 Code Generator Options 1167
16.13 SPL (Spy Programming Language) 1168
16.13.1 Basic SPL structure 1168
16.13.2 Declarations 1169
16.13.3 Variables 1170
16.13.4 Predefined variables 1171
16.13.5 Creating output files 1172
16.13.6 Operators 1173
16.13.7 Conditions 1174
16.13.8 Collections and foreach 1175
16.13.9 Subroutines 1176
16.13.10 Built in Types 1179

17 The MapForce API 1186

17.1 Overview 1187
17.1.1 Accessing the API 1187
17.1.2 The Object Model 1189
17.1.3 Error Handling 1189
17.1.4 Examples 1191
17.2 Object Reference 1212
17.2.1 Application 1212
17.2.2 AppOutputLine 1220
17.2.3 AppOutputLines 1225
17.2.4 AppOutputLineSymbol 1226
17.2.5 Component 1227
17.2.6 Components 1234
17.2.7 Connection 1235
17.2.8 Datapoint 1236
17.2.9 Document 1237
17.2.10 Documents 1246
17.2.11 ErrorMarker 1248
17.2.12 ErrorMarkers 1250
17.2.13 MapForceView 1252
17.2.14 Mapping 1256

17.2.15 Mappings ... 1260
17.2.16 Options ... 1261
17.2.17 Project ... 1266
17.2.18 ProjectItem ... 1274
17.3 Enumerations ... 1282
17.3.1 ENUMApacheAxisVersion (obsolete) ... 1282
17.3.2 ENUMApplicationStatus ... 1282
17.3.3 ENUMAppOutputLine_Severity ... 1282
17.3.4 ENUMAppOutputLine_TextDecoration ... 1283
17.3.5 ENUMCodeGenErrorLevel ... 1283
17.3.6 ENUMComponentDatapointSide ... 1283
17.3.7 ENUMComponentSubType ... 1284
17.3.8 ENUMComponentType ... 1284
17.3.9 ENUMComponentUsageKind ... 1284
17.3.10 ENUMConnectionType ... 1284
17.3.11 ENUMDOMType ... 1285
17.3.12 ENUMLibType ... 1285
17.3.13 ENUMProgrammingLanguage ... 1285
17.3.14 ENUMProjectItemType ... 1286
17.3.15 ENUMProjectType ... 1286
17.3.16 ENUMSearchDatapointFlags ... 1287
17.3.17 ENUMViewMode ... 1287

18 ActiveX Integration 1290

18.1 Prerequisites ... 1291
18.2 Adding the ActiveX Controls to the Toolbox ... 1293
18.3 Integration at Application Level ... 1295
18.4 Integration at Document Level ... 1297
18.5 ActiveX Integration Examples ... 1301
18.5.1 C# ... 1301
18.5.2 HTML ... 1307
18.5.3 Java ... 1314
18.5.4 VB.NET ... 1322
18.6 Command Reference ... 1325
18.6.1 "File" Menu ... 1325
18.6.2 "Edit" Menu ... 1326

18.6.3 "Insert" Menu ... 1326
18.6.4 "Project" Menu ... 1327
18.6.5 "Component" Menu ... 1328
18.6.6 "Connection" Menu ... 1329
18.6.7 "Function" Menu ... 1330
18.6.8 "Output" Menu ... 1330
18.6.9 "Debug" Menu ... 1331
18.6.10 "View" Menu ... 1331
18.6.11 "Tools" Menu ... 1332
18.6.12 "Window" Menu ... 1332
18.6.13 "Help" Menu ... 1332
18.7 Object Reference ... 1334
18.7.1 MapForceCommand ... 1334
18.7.2 MapForceCommands ... 1336
18.7.3 MapForceControl ... 1337
18.7.4 MapForceControlDocument ... 1344
18.7.5 MapForceControlPlaceHolder ... 1350
18.7.6 Enumerations ... 1353

19 付録 1356

19.1 エンジン情報 ... 1357
19.1.1 XSLT および XQuery エンジンに関する情報 ... 1357
19.1.2 XSLT と XPath/ XQuery 関数 ... 1362
19.2 技術データ ... 1438
19.2.1 OS とメモリ要件 ... 1438
19.2.2 Altova XML バリデーター ... 1438
19.2.3 Altova XSLT と XQuery エンジン ... 1438
19.2.4 Unicode のサポート ... 1439
19.2.5 インターネットの使用 ... 1439
19.3 ライセンス情報 ... 1440
19.3.1 電子的なソフトウェアの配布 ... 1440
19.3.2 ソフトウェアのアクティベーションとライセンスの計測 ... 1441
19.3.3 知的所有権 ... 1441
19.3.4 エンドユーザー使用許諾契約書 ... 1442

20 用語 1454

20.1 C 1455
20.2 F 1456
20.3 I 1457
20.4 J 1458
20.5 M 1459
20.6 O 1460
20.7 P 1461
20.8 S 1462
20.9 T 1463

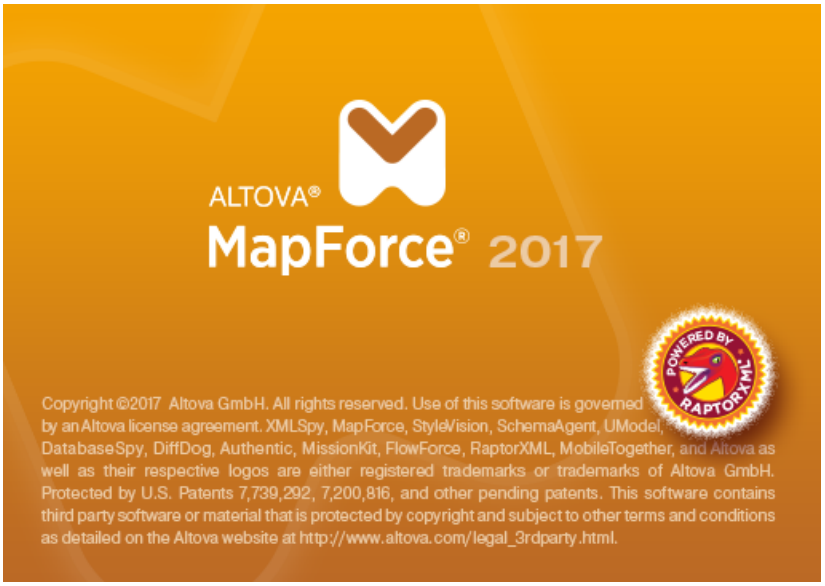
インデックス

Chapter 1

Altova MapForce 2017 Enterprise Edition

1 Altova MapForce 2017 Enterprise Edition

MapForce® 2017 Enterprise Edition は高度なデータ統合プロジェクトにて決定的な役割を果たす視覚的なデータマッピングツールです MapForce(R) Windows XP/Vista, Windows 7/8/10 と Windows Server 2003/2008/2012/2016 で作動します。



最終更新日：201?年 04月 28日

1.1 新規機能

MapForce 2017 リリース 3の新規機能:

- 与えられた条件に基づき、2つの異なる構造からのデータをジョインするために使用する新規のコンポーネント型(ジョイン)が搭載されるようになりました (次を参照 :データのジョイン)。マッピングがデータをデータベースから読み取る時、データベーステーブルをジョインすることができ、また、SQL JOIN モードでビューすることができます。次を参照してください:データベースデータのジョイン (SQL JOIN)。
- JSON5 構文を使用するJSON ファイルからのマッピング、JSON ファイルへのマッピングへのサポート (次を参照 :JSON5 へのサポート)。
- 「出力」ペイン、「Query」ペイン、および「SLT」ペイン内でのテキストの検索オプション。が強化されました (次を参照 :テキストビュー内の検索)。また、テキストのハイライトも上記のペインで使用することができます (次を参照 :テキストのハイライト)。
- データベースを更新するマッピングが、NULL 対応しデータを比較するため、任意で構成できるようになりました。NULL 対応する比較は、null 値を含むデータを (各特殊のデータベースに合わせて) よりよく扱う方法を提供します (次を参照 :データベーステーブルアクション内で Null を扱う)。
- MapForce ActiveX コントロール内で、`MapForceCommand` オブジェクトの構造がコマンドの一意の名前を取得するために使用することのできる新規の `Name` プロパティを含むように強化されました。これにより、MapForce コマンドのプログラミング上の情報の取得が簡素化されました (次を参照 :コマンド情報の抽出)。
- 内部のアップデートと最適化

MapForce 2017 の新規機能:

- マッピングは、ネイティブな接続を使用してPostgreSQL データベースに接続することができます (次を参照してください:PostgreSQL 接続のセットアップ)。
- マッピングは、ADO.NET プロバイダを使用してSQL Server と他のデータベース型に接続することができます。(次を参照してください:ADO.NET 接続のセットアップ)
- 新しいデータベース型へのサポート:Progress OpenEdge。次を参照してください:Progress OpenEdge (ODBC) へ接続するとProgress OpenEdge (JDBC) へ接続する。
- JDBC を使用してデータベースに接続する場合、jar ライブラリへの検索パスをデータベース接続ダイアログで直接指定することができます (次を参照してください:JDBC 接続のセットアップ)
- XBRL タクソノミパッケージのためのサポート
- インラインXBRL 読み取りのためのサポート
- Visual Studio のためのMapForce プラグインとEclipse のためのMapForce プラグイン内でマッピングのデバッグを使用することができます。
- EDIFACT D.15B とD.16A ディレクトリのためのサポート (次を参照してください:EDI)。D.16A ディレクトリは、大量のメッセージの検証 (VERMAS) へのサポートを含みます。
- EDI メッセージの検証は、構成ファイルを使用してカスタム化することができます。これにより、検証ルーツをリラックスさせる、または、EDI 標準により参照されるMapForce 外部コードの値に適用することができます。 (次を参照してください:構成ファイルを使用してEDI 検証をカスタマイズする)。
- 「Update if… Insert Rest」アクションを使用したマッピングによりデータベースがアップデートされる場合、MERGE ステートメントが選択されたデータベースのために作成されます (次を参照してください:MERGE ステートメント)。
- MapForce Server 実行ファイル (mfx) を、特定 MapForce Server のバージョンのためにコンパイルすることができます (次を参照してください:特定のMapForce Server バージョンのためにマッピングをコンパイルする)
- ソースXML からノード名を読み取る (または、CSV/テキスト ファイルからフィールド名) を読み取ることは可能でターゲットにこの情報をマップすることができます。また、動的に新規のXML 属性、または、要素 (または、CSV/Text フィールド名) をソースから与えられた値をベースにしたターゲット内に作成することもできます。ノード名のマッピング を参照してください。

- 要素レベルでカスタム化された名前空間を持つXML インスタンスファイルを作成することができます (次を参照してください:カスタム名前空間の宣言)
- 内部のアップデートと最適化

MapForce 2016 R2 の新規機能:

- XSLT ペイン内の更に直感的なコードの使用 :折りたたまれたテキストが、省略シンボルと共に表示され、ツールヒントとして、プレビューすることができます。XQuery ペインとSQL エディター内のテキストにも同じルールが適用されます
- アクティブなマッピング内で関数が使用される全ての箇所を検索することができます (ライブラリ ウィンドウ 内の関数を右クリックし、 **全ての呼び出しを検索する** を選択します)。
- - スタイルのWeb サービスをマッピングから呼び出すことができます。次を参照してください : Web サービスの呼び出しの追加 (REST- style)。 WSDL- style Web サービスのための既存のサポートへの追加
- EDIFACT 2015A ディレクトリへのサポート (次を参照してください:EDI)
- Eclipse 4.5 へのためのサポート (次を参照してください:Eclipse のための MapForce プラグイン)
- 内部のアップデートと最適化

MapForce 2016 の新規機能 :

- XSLT 1.0 コードの改善された生成 (生成されたスタイルシートは読み込みやすく、より早く実行することができます)。
- 新規 集計関数が MapForce コアライブラリで使用することができます **`min-string`** と **`max-string`**。これらの関数は文字列のシーケンスから最大値および最小値を取得することができます。
- 内蔵の実行エンジンのためにかかれたマッピングはデバックすることができます。 (マッピングのデバッグを参照してください)。
- Visual Studio のための MapForce プラグイン は Visual Studio 2015 (前のバージョンへのサポートに追加して)をサポートします。
- (前のバージョンへのサポートに追加して)新しいデータベースバージョンがサポートされます :SQL Server 2014、Oracle 12c、IBM DB2 10.5、PostgreSQL 9.4、MySQL 5.6。
- Firebird データベースはサポートされています。 (Firebird (ODBC) への接続 と Firebird (JDBC) への接続を参照してください。)

MapForce バージョン 2015 R4 の新規機能 :

- TRADACOMS フォーマットへ、またはからのマッピングのデータをサポートします
- Eclipse のための MapForce プラグイン。MapForce ファイル 特有のコマンドは 新規 **MapForce** メニューで使用することができます。 (共通メニューと関数へのアクセスを参照してください)。
- 内部的な更新と最適化

MapForce バージョン 2015R3 の新規機能 :

- XML 出力内の `<?xml ... ?>`宣言を抑制するオプション
- Web サービスを呼び出す際、プリエンプティブ HTTP 認証、HTTPS、と基本的 WS- Security を使用することができます (HTTP セキュリティの設定 と WS- Security の設定を参照してください)。
- Support for reading Strict Open XML Excel 2013 ファイル
- FlexText は正規表現を使用して分割および切り替えを行います。 (FlexText と正規表現を参

照してください）。
- (EDI、CSV、固定値フィールド、JSON、XML を含む）テキストベースのコンポーネントは、プレーンファイルに加え、文字列を解析およびシリアル化することができます。
- SQLite データベースへのサポート（SQLite 接続のセットアップを参照してください）。
- 新規文字列パディング関数：pad-string-left と pad-string-right
- 新しいコンポーネント型：単純型出力
- 内部的な更新と最適化

MapForce バージョン 2015 の新規機能：

- 新規 language 引数は format-date と format-dateTime 関数で使用することができます
- 新規シーケンス関数：replicate-item
- XBRL Table Linkbase へのサポート
- FlexText「一度分割」オプションのための新しいモード：カンマ区切り（ラインの開始）
- JSON ファイルへまたはからのマップ

MapForce バージョン 2014R2 の新規機能：

- 新規シーケンス関数: シーケンスの生成、item-at など。
- 出力コンポーネント内の CDATA セクションを定義する機能
- データベースの実行と Web サービス呼び出しのためのタイムアウトの値を定義する機能
- コンポーネントの削除後接続を保存する方法
- データベースデータのバルク転送（全てをバルク挿入）
- ターゲットコンポーネント内の必須アイテムの自動ハイライト

MapForce バージョン 2014 の新規機能:

- RaptorXML 検証への統合と XML Schema 1.1 への基本的なサポート
- 新規 RaptorXML XSLT と XQuery エンジンの統合
- XML スキーマワイルドカードへのサポート、xs:any と xs:anyAttribute
- XML ターゲットコンポーネント内のコメントと処理命令へのサポート
- 年齢関数
- CSV データのため引用文字を挿入

MapForce バージョン 2013R2 SP1 の新規機能:

- 高速変換エンジン RaptorXML Server

MapForce バージョン 2013R2 の新規機能:

- MapForce Server のサポート。
- コマンドラインやファイルメニューから MapForce Server 実行ファイルを生成することで、MapForce Server にて実行することができます。
- MapForce のマッピングを FlowForce Server へデプロイすることができます。
- Informix 11.7 データベースのサポートと、その他のデータベースのサポートの拡張。
- 出力ファイルで使用される行末のユーザー定義。ユーザー定義。
- 内部的な更新と最適化

MapForce バージョン 2013 の新規機能:

- マッピングにてストアドプロシージャの呼び出し
- データベース関数のサポート (ストアドプロシージャに類似した機能)
- パラメーターを伴うSELECT ステートメントのサポート
- 内部的な更新と最適化

MapForce バージョン 2012R2 の新規機能:

- XSLT 2.0、XQuery、内蔵の実行エンジンにて ソートコンポーネント をサポート
- ユーザー定義コンポーネント名
- SQL-WHERE 機能の拡張 :ORDER BY
- IBM iSeries のデータベースに対応した論理ファイルをサポート:論理ファイルがビューとして表示されます
- IBM DB2 論理ファイルのサポート:DB2 データベースのIBM iSeries エディションに搭載された論理ファイルにより、1つまたは複数の物理ファイルが表されます。論理ファイルによりシーケンシャルなかたちでデータへアクセスすることができるほか、物理ファイルとは異なるフォーマットを行うことができます。IBM iSeries コンピューターへ接続すると、論理ファイルにより構成された既存のデータベースを扱うことになる場合もあります。このようなデータへはこれまでアクセスすることができませんでしたが、バージョン 2012 リリース 2 からはそれが可能になりました。

MapForce バージョン 2012 の新規機能:

- XML、CSV、FLF ファイルに対してデータストリーミングを内蔵の実行エンジンにて処理
- 新たなデータベースエンジンによるODBC ならびにJDBC 接続のサポート。
- ウィンドウマッピングウィンドウにおけるコンポーネントの自動配置
- 新しい関数 :parse-date とparse-time
- プロジェクトタブウィンドウからのアイテムの検索
- ターゲット親 ノードへの接続を通知
- マッピングにてコンポーネントのシーケンスを制御するルール
- MapForceAPI のセクションに掲載されている一新されたプログラミング言語のサンプル

MapForce バージョン 2011R3 の新規機能:

- 中間変数
- Excel コンポーネントにて、複数の行領域をサポート
- HIPAA の実装ガイドであるEDI X12 のサポート
- X12 EDI 999 Implementation Acknowledgement コンポーネントの生成
- XML デジタル署名 のサポート
- Support for US-GAAP 2011
- .NET Framework 4.0 アセンブリファイルのサポート
- コマンドラインからStyleVision 形式でフォーマットされたドキュメントを生成

MapForce バージョン 2011R2 の新規機能:

- ビルトイン実行エンジンによるストリーム出力サポート
- ライブラリウィンドウライブラリウィンドウにおける検索機能
- 反転 マッピング
- 拡張可能なIF-ELSE 関数
- Core ライブラリにおけるノード名ならびに解析関数
- 新たに追加されたEDI フォーマットのIATA PADIS
- コンポーネント毎に複数のEDI メッセージ を処理
- キー生成設定の統合により改善されたデータベーステーブルアクションダイアログ

- マッピングの ドキュメンテーション生成 を行う際にStyleVision Power Stylesheet を使用するオプションを追加

MapForce バージョン 2011 の新規機能:

- マッピングチェーンにて、中間コンポーネントのプレビュー (パススループレビュー)
- サポートされている全ての言語において dateTime とnumbers のフォーマット機能が利用可能に
- 自動連番 機能の強化
- remove-timezone とconvert-to-utc のタイムゾーンに関する関数を新たに追加
- StyleVision チャートを含むStyleVision Power Stylesheets を使用するターゲットコンポーネントのプレビュー

MapForce バージョン 2010 リリース 3 の新規機能:

- C# ならびに C++ において Visual Studio 2010 プロジェクトファイルの生成をサポート
- ワークシート行を Excel コンポーネントにおける列名として定義可能に
- 生成されたC++ コードにてMSXML 6.0 をサポート
- Nillable 値 ならびにXML インスタンスファイルにおけるxsi:nil 属性をサポート
- XML ドキュメントにてターゲットに対する自動キャスティングを無効化
- SAP IDocs のサポート

MapForce バージョン 2010 リリース 2 の新規機能:

- 64 ビットオペレーティングシステム (Windows Server 2003/2008、Windows XP、Windows Vista ならびにWindows 7) 上で64 ビットバージョンのMapForce Enterprise / Professional エディションをサポート7
- Excel 2010 のサポート
- 親コネクタの接続時に、対応する子接続を自動的に接続
- SQL Where コンポーネントにてフィールドのサポート
- compiled Java .class と .NET アセンブリファイルを追加可能に
- 入力文字列の トークン化により更なる処理
- UN/ EDIFACT ならびに ANSI X12 EDI ソースファイルの検証
- X12 EDI 997 Functional Acknowledgement コンポーネントの生成

MapForce バージョン 2010 の新規機能:

- コンポーネント毎に複数の入力 /出力サポート
- 相対パス サポートのアップグレード
- xsi:type のサポートによる派生型の使用
- 新たに追加されたデータ型システム
- ユーザー定義の関数ナビゲーションの改善
- 生成されたコードにおけるEDI 出力の検証 (次を参照 :EDI コンポーネントの検証)
- EDIFACT サービスメッセージであるCONTRL、AUTACK、そしてKEYMAN のサポート
- WSDL 2.0 により定義されたWeb サービスのサポート
- XML 要素における複合コンテンツの処理を強化

MapForce バージョン 2009SP1 の新規機能:

- ユーザー定義関数における パラメーターの順序を定義
- XML スキーマに対して妥当ではない XML ファイルの処理
- 全般的 (標準的) なユーザー定義関数における複雑な階層構造の引数サポート

- C++ コードの生成時における Apache Xerces 3.x のサポート

MapForce バージョン 2009 の新規機能:

- XBRL のサポートにより、XBRL 次元インターフェースファイルならびに XBRL タクソノミファイルを、ソースとターゲットコンポーネントとして使用可能に
- EDI HL7 バージョン 2.2 から 2.6 コンポーネントをソースならびにターゲットコンポーネントとして使用可能に
- EDI HL7 versions 3.x XML をソースならびにターゲットコンポーネントとして使用可能に
- マッピングプロジェクトのドキュメンテーション
- SQL Server の XML フィールドをネイティブサポート
- ノードコンテンツやノードのグループ化
- シーケンス内の ノードの位置 をベースにしたデータのフィルタリング
- QName サポート
- コンポーネント内にあるアイテム/ノードの検索

MapForce バージョン 2008 リリース 2 の新規機能:

- XML ファイルのために XML スキーマを自動生成
- Office Open XML Excel 2007 または以降 (*.xslx) をソースならびにターゲットコンポーネントとしてサポート
- 生成された Java ならびに C# コードの入出力として ストリームオブジェクト をサポート
- C++ ならびに C# における Visual Studio 2008 プロジェクトファイルの生成
- Web サービスにて SOAP バージョン 1.2 をサポート
- FlexText にて、連続分割オプションの "先頭" を新たにサポート
- 生成されたコードからデータベーススキーマ名を除外するオプション
- データベースコンポーネント内の視覚的なテーブルとして用意された SQL SELECT ステートメント
- ローカルリレーション - - 主キー外部キーリレーションシップを素早く作成
- Altova グローバルリソース のサポート
- パフォーマンスの最適化

MapForce バージョン 2008 の新規機能:

- 集計 関数
- Value-Map 検索コンポーネント
- XML 出力オプションの強化 XML 出力の整形、個々のコンポーネントに対してのエンコーディング設定や XML スキーマ参照の省略
- 内部における様々なアップデート

MapForce バージョン 2007 リリース 3 の新規機能:

- データベースフィールドへ/からの XML データマッピング (XML データをデータベースフィールドへ /からマッピング を参照してください)。
- データベースへの直接のクエリ
- SQL-WHERE フィルターならびに SQL ステートメントウィザード
- コードジェネレータの最適化ならびにドキュメンテーションの改善
- EDI X12 リリースの 3040 から 5030 を全てサポート
- 93A から 08B の UN/ EDIFACT メッセージを完全にサポート

Chapter 2

はじめに

2 はじめに

MapForce の機能、ユーザーインターフェイス、MapForce の基本概念、およびこのドキュメントで使用される規約を紹介します。

2.1 MapForce とは ?

MapForce は、プログラムコードを作成することなく視覚的なドラッグアンドドロップを使用するグラフィカルなユーザーインターフェイスを用いデータを異なるデータ間、スキーマ間で変換するWindows ベースの多目的のIDE (統合された開発環境 統合開発環境) です。また、事実上、MapForce は時際のデータ変換 (またデータマッピング)を行うプログラムコードを生成します。プログラムコードを生成しない場合、 (MapForce Professional またはEnterprise Editionsで使用することのできる)MapForce 内蔵の変換言語 を実行します。

MapForce を使用して、便利にファイルベースおよび他のフォーマットの多種のデータへ、またはこれらからからデータを変換することができます。使用する技術に関わりなく、MapForce は自動的にデータの構造を決定し、データのためのスキーマを提供するオプションを提供し、サンプルインスタンスファイルから自動的に生成します。例 :例えば、XML インスタンスファイルがあり、スキーマ定義がない場合 MapForce はスキーマを、XML ファイル内で使用することのできるデータを他のファイルまたはフォーマットでマッピングのために使用できるように生成することができます。

マッピングソースまたはターゲットとしてサポートされるテクノロジーは、以下のとおりです。

MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
• XML とXML スキーマ • HL7 バージョン 3x (schema-based)	• XML とXML スキーマ • コンマで区切られた値 (CSV) と固定長フィールドフォーマット (FLF) を含むフラットファイル • データベース (Microsoft Access や SQLite データベースなどを含む全てのメジャーなデータベース)	• XML とXML スキーマ • コンマで区切られた値 (CSV) と固定長フィールドフォーマット (FLF) を含むフラットファイル • マップし、MapForce FlexText を使用して、他のフォーマットに変換することのできるレガシーテキストファイルからのデータ。 • データベース (Microsoft Access や SQLite データベースなどを含む全てのメジャーなデータベース) • (UN/EDIFACT、ANSI X12、HL7、IATA PADIS、SAP IDoc、TRADACOMSなどを含む)EDI フォーマットのファミリー • JSON ファイル • Microsoft Excel 2007 と以降のファイル • XBRL インスタンスファイルとタクソノミ

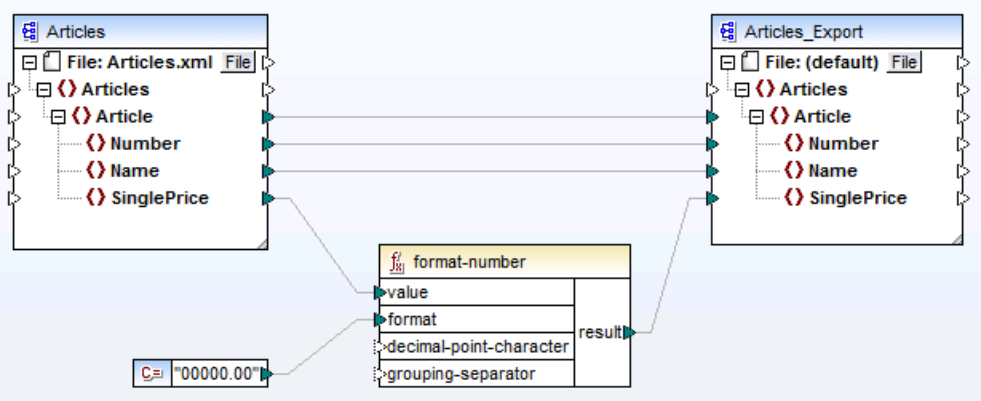
MapForce エディションにより、データの変換言語を以下のように選択することができます。

MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
• XSLT 1.0 • XSLT 2.0	• MapForce 内蔵の変換言語 • XSLT 1.0 • XSLT 2.0 • XQuery • Java • C# • C++	• MapForce 内蔵の変換言語 • XSLT 1.0 • XSLT 2.0 • XQuery • Java • C# • C++

全ての変換の結果、および生成された XSLT または XQuery コードをグラフィカルユーザーインターフェイスを移動することなくをプレビューすることができます。デザインまたはマッピングをプレビューする際、MapForce はスキーマおよび変換の整合性を検証し、検証エラーを専用のウィンドウで表示するため、すぐに確認して、エラーの修正を行うことができます。

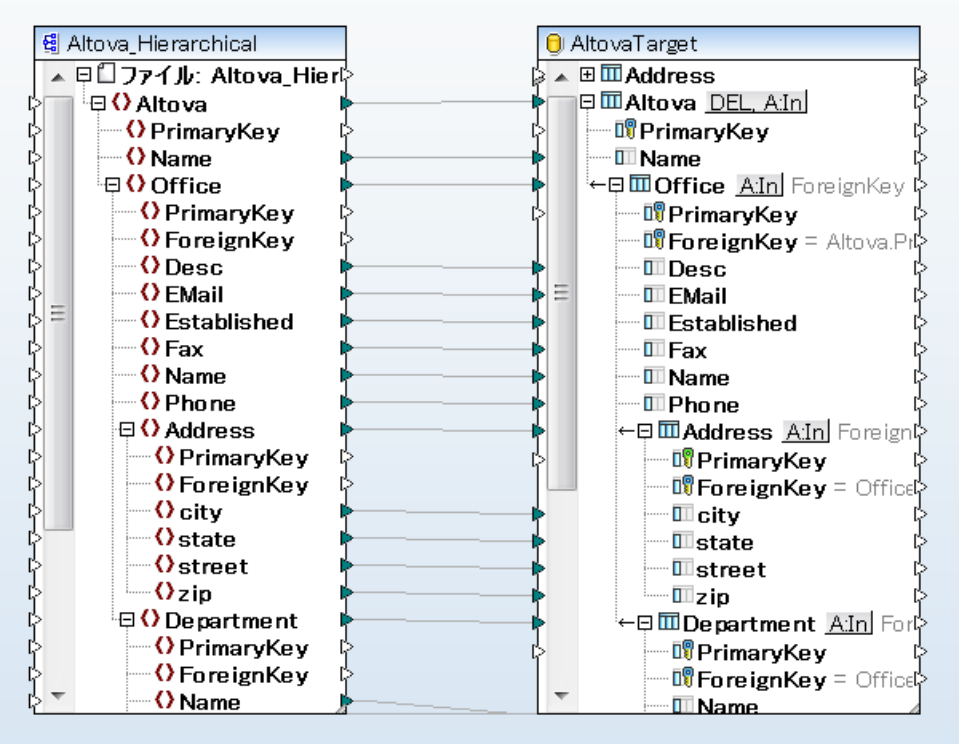
Java、C#、または C++ を変換言語として選択する場合、MapForce は必要とされるプロジェクトとソリューションを生成し、Visual Studio または Eclipse で直接開き、生成されたデータマッピング プログラムを実行することができます。高度なデータ統合のシナリオの場合、自身のコードを用い Altova と MapForce API を使用して、生成されたプログラムを拡張します。

MapForce では、全てのマッピングの変換を視覚的にデザインすることができます。例 : XML の場合、全ての要素と属性を接続することができます、または要素または 属性 他の XML ファイルの要素または属性の XML ファイルにコメントを挿入します。これにより、MapForce にソース要素 (または属性)からデータを読み込み、ターゲット要素 (または 属性)に書き込むことを命令します。



2 つのXML ファイル間のサンプルデータ変換

同様に、MapForce Professional または Enterprise Editions でデータベースと作業する場合、MapForce のマッピングエリア内のデータベースカラムを確認し、データをマップし、視覚的な接続を作成することができます。Altova MissionKit 製品と同様、MapForce からデータベース接続を設定する場合、柔軟的にデータベースドライバーと接続の型 (ADO、ODBC、または JDBC) を既存のインフラストラクチャとデータマッピングの必要に応じて選択することができます。更に、 SQL クエリを視覚的に作成し、ストアドプロシージャを使用し、または、データベースに直接問い合わせることができます。 (データベース型、エディション、ドライバーによりサポートは異なります)



XML ファイルとデータベース間のサンプルデータ変換

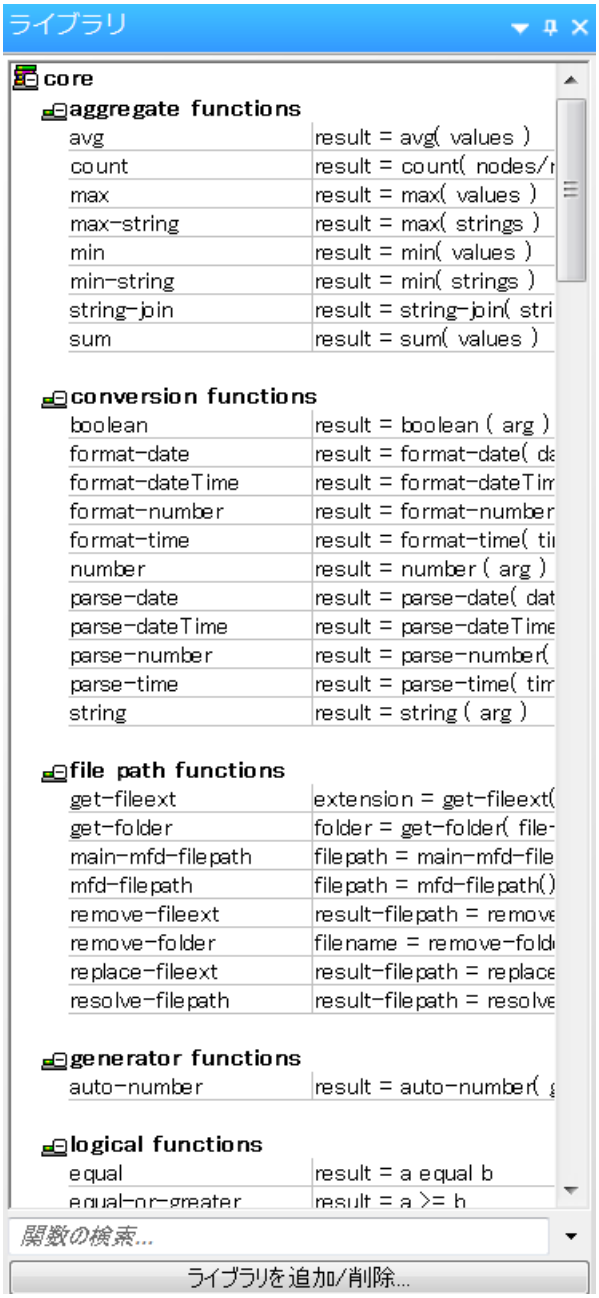
MapForce を使用したマッピングデザインのとても簡単な例として、 「ソースX からデータを読み込み ターゲットY に書き込む"が挙げられます。しかしながら、 「ソースX からデータを読み込み、 ターゲットY に書き込む。そしてソースY からデータを読み込み ターゲットZ へ書き込む"などのMapForce シナリオを簡単にデザインすることができます。これらは 「パススルー"または 「チェーン"マッピングとして知られ、これらにより変換プロセスの中間のステージでデータにアクセスすることができます (例えばファイルとして保存するため)。

MapForce 内で作成することのできるデータマッピングは、単一かつ定義済みファイルだけではありません。同じ変換内で、ディレクトリから複数の入力ファイルを動的に処理することができ、複数の出力ファイルを生成することができます。ですから、 「複数のX ファイルからデータを読み込み、単一のY ファイルに書き込む"または "ファイルX を読み込み複数の ファイルY を生成する"といったシナリオが可能になります。

重要な点は、同じ変換で、使用中のMapForce エディションでサポートされる全てのデータ型の複数の ソースと複数のターゲットを混合できる点です。例 : MapForce Professional またはEnterprise の場合、2つの異なるデータベースを単一のXML ファイルにマージすることが可能です。または、複数のXML ファイルからのデータをマージし、一部のデータを1つのデータベースとして書き込み、一部のデータを他のデータベースとして書き込むことができます。SQL ステートメントをデータベースにコミットする前にプレビューすることができます。

ソースからターゲットへのデータの直接の変換は通常の変換目的ですが、多くの場合、データを特定の方法で処理する必要がある場合があります (例 :並べ替え、グループ化、または フィルター) などを最終プロセスの前に行う必要があります。この理由のため、MapForce は、プログラミング言語の構成を簡素化するその他の関数コンポーネント (規則、変数、SQL-WHERE 条件、フィルターと並べ替えコンポーネントなど) を搭載しています。また、MapForce は、全てのデータ操作をアシストする豊富かつ広範囲の関数ライブラリも搭載しています。

必要であれば、ビルドインライブラリを、MapForce で直接関数 (いわゆるユーザー定義関数、またはUDF) をデザインすることにより、またはXSLT、XQuery、Java、 またはC# 言語で外部的に作成された関数やライブラリを使用して拡張することも可能です



ライブラリペイン (MapForce Basic Edition)

データマッピングデザインファイルの数が多すぎる場合、マッピングをプロジェクトに整理することができます (MapForce Professional と Enterprise エディションで使用することができます)。これによりより簡単なアクセスと管理を行うことができます。重要な点は、プロジェクト内の個別のマッピングのためにコードを生成し、更にプログラムコードをプロジェクト全体から生成することができます。

(MapForce Server API を使用したマッピング変換の実行時などの) 高度なデータ処理のために マッピングをデザインし、ランタイムで値をパス、またはランタイムから単純型文字列の値を取得することができます。この機能は、関数または単純型文字列の値を生成するマッピング全体を素早くテストすることを可能にします。MapForce の Professional と Enterprise エディションは、他のプログラミング言語内で同様に動作する、ランタイムでの文字列の解析とシリアル化を行うコンポーネントも含みます。

MapForce Enterprise Edition では、Web サービス言語定義 (WSDL) ファイルをベースにした SOAP 1.0 と SOAP 2.0 ウェイブサービスを視覚的にデザインすることができます。WSDL 1.0 または WSDL 2.0 Web サービスからデータをマッピング内部から取得することができます。これは、HTTP と HTTPS プロトコールを使用して、ユーザーが WS- Security メカニズムまたは HTTP 認証を使用することを必要とするWeb サービスで使用することができます。

MapForce Professional と Enterprise Editions では、マッピングデザイン ファイルの詳細なドキュメントを HTML、Word 2007+ または RTF フォーマットで生成することができます。 例 :ドキュメントから特定のコンポーネントを包含したり削除することが選択可能です)。

MapForce を他の Altova MissionKit 製品と併せて使用している場合、MapForce は、これらの製品と Altova サーバーベースの製品を以下のテーブルに表示されているように統合します。

MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
生成された XSLT を直接 MapForce 内で実行し、データ変換の結果をすぐに確認することができます。パフォーマンスを向上するには、超高速 XML 変換エンジン RaptorXML Server を使用して、マッピングを処理します。		
XMLSpy がインストールされている場合、同じマシンで、関連する MapForce コンテキストから直接 XMLSpy を開き、サポートされるファイル型を便利に開き編集することができます。 (例 : **「コンポーネント | XMLSpy 内でスキーマ定義を編集」** メニューコマンドは XML コンポーネントをクリックすると使用することができます)。		
	データ変換を、直接 MapForce 内で、またはコマンドラインや自動化された実行を使用して異なるマシンやオペレーティングシステムにデプロイし実行することができます。 具体的な方法は、Windows 上でマッピングをデザインし、MapForce Server または FlowForce Server が動作する Windows、Linux、または Mac サーバーマシンで実行することができます。
	StyleVision が同じマシンにインストールされている場合、デザインし、既存の ,StyleVision スタイルシートを再利用し、マッピングの結果を HTML、RTF、PDF、または Word 2007+ ドキュメントとしてプレビューすることができます。

MapForce Professional と Enterprise エディションは、Visual Studio と Eclipse の統合された開発環境のプラグインとしてインストールすることができます。これにより、優先する開発環境を移動することなく、MapForce の機能にアクセスしマッピングをデザインすることができます。

MapForce では、開発環境の概観や使用感を完全にカスタム化できるだけでなく (グラフィカルユーザーインターフェイス)、各技術と各マッピングのコンポーネントの型の他の関連する設定 もカスタム化することができます。例 :

- XML から、または、XML へ マッピングを行う際、スキーマレファレンスを含むか否かを選択することができ、または XML 宣言 が XML 出力ファイル内で抑制されるべきかを選択することができます。生成されたファイルのエンコードを選択することができます (例 :UTF-8
- データベースから、または、データベースへマッピングする際、データベース ステートメントの実行のタイムアウト期間、MapForce がデータベーストランザクションを使用するか、または、コード生成の際データベーススキーマ名からテーブル名を削除するかなど、などの設定を定義することができます。
- XBRL の場合、MapForce が表示する構造ビューを選択することができます (「プレゼンテーション定義リンクベース"ビュー、 「テーブルリンクベース」ビュー、または 「全てのコンセプト」ビューなど)。

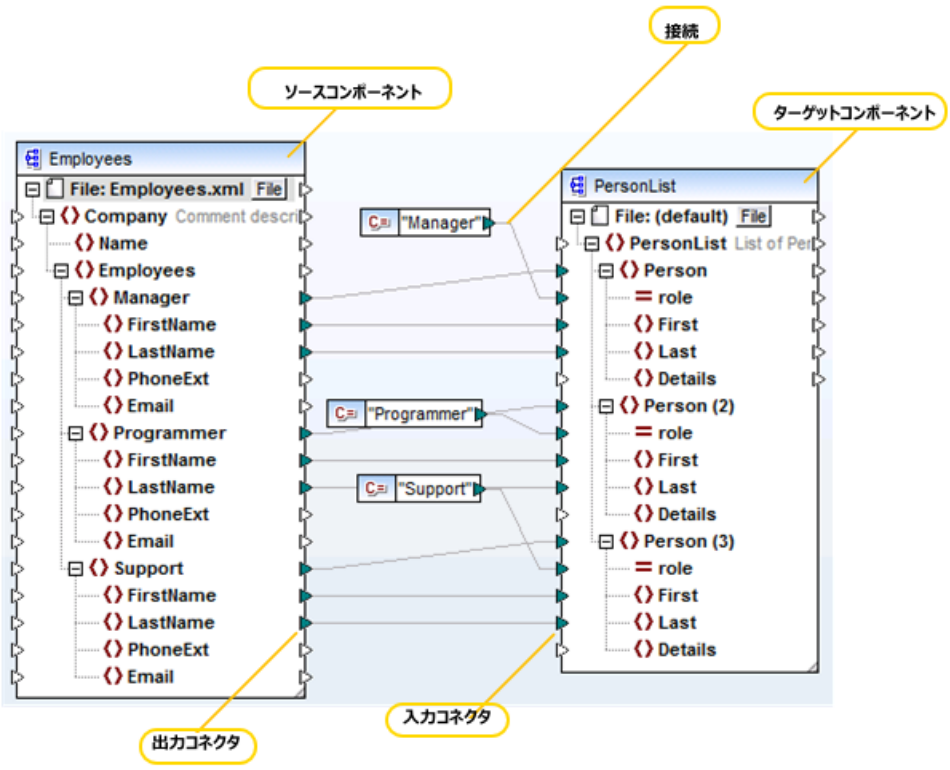
MapForce の全てのエディションは 32-ビットアプリケーションで動作します。更に、MapForce Professional と Enterprise エディションは 64-ビットアプリケーションで動作します。

2.2 基本概念

このセクションはデータマッピングを開始するにあたり必要な基本概念のアウトラインを紹介します。

マッピング

MapForce マッピングデザイン (または、略して "マッピング") は、データがどのように1つのフォーマットから他のフォーマットに変換されるかの視覚的な表現です。マッピングは、データ変換を行うためのMapForce マッピングエリア内で1つのスキーマから他のスキーマに追加するコンポーネントから構成されています。 (例 :XML ドキュメントを1つのスキーマから他のスキーマに変換)。有効なマッピングは、1つまたは複数のターゲットコンポーネントに接続されている、1つまたは複数の複数のソースコンポーネントから構成されています。マッピングを実行して、結果を直接 MapForce 内で確認することができます。コードを生成し、外部で実行することも可能です。MapForce 実行可能ファイルにマッピングをコンパイルし、MapForce Server またはFlowForce Server を使用してマッピングの実行を自動化することもできます。 MapForce は、マッピングを .mfd の拡張子を持つファイルとして保存します。



MapForce マッピングの基本的構造

コンポーネント

MapForce 内では、 "コンポーネント" という用語はデータの構造 (スキーマ) またはデータがどのように変換 (関数) されるかを視覚的に表します。コンポーネントはすべてのマッピングを構築する中心的な役割を果たします。以下は、MapForce コンポーネントの例です :

- 定数

- データベース
- フィルター
- 条件
- フラットファイル (CSV、固定長と他のテキストファイル)
- 関数コンポーネント
- EDI ドキュメント (UN/EDIFACT、ANSI X12、HL7)
- Excel 2007+ ファイル
- 単純型 入力コンポーネント
- 単純型 出力コンポーネント
- 変数
- XBRL ドキュメント
- スキーマおよびDTD

コネクタ

コンポーネントの左右に表示される小さな三角形です。コンポーネントの左に表示されるコネクタはそのコンポーネントのエントリポイントを表します。コンポーネントの右に表示されるコネクタはそのコンポーネントからのデータの終了ポイントを表します。

接続

接続とは2つのコネクタ間に描くことができる線です。接続を描くことにより、MapForce にデータを特定の方法で変換するように命令します。(例 :XML ドキュメントからデータを読み込み、他のXML ドキュメントに書き込みます)。

ソースコンポーネント

ソースコンポーネントとは、MapForce がデータを読み込む元のコンポーネントです。マッピングを実行すると、ソースコンポーネントのコネクタにより与えられたデータを読み込み、必要とされる型に変換し、ターゲットコンポーネントのコネクタに送信します。

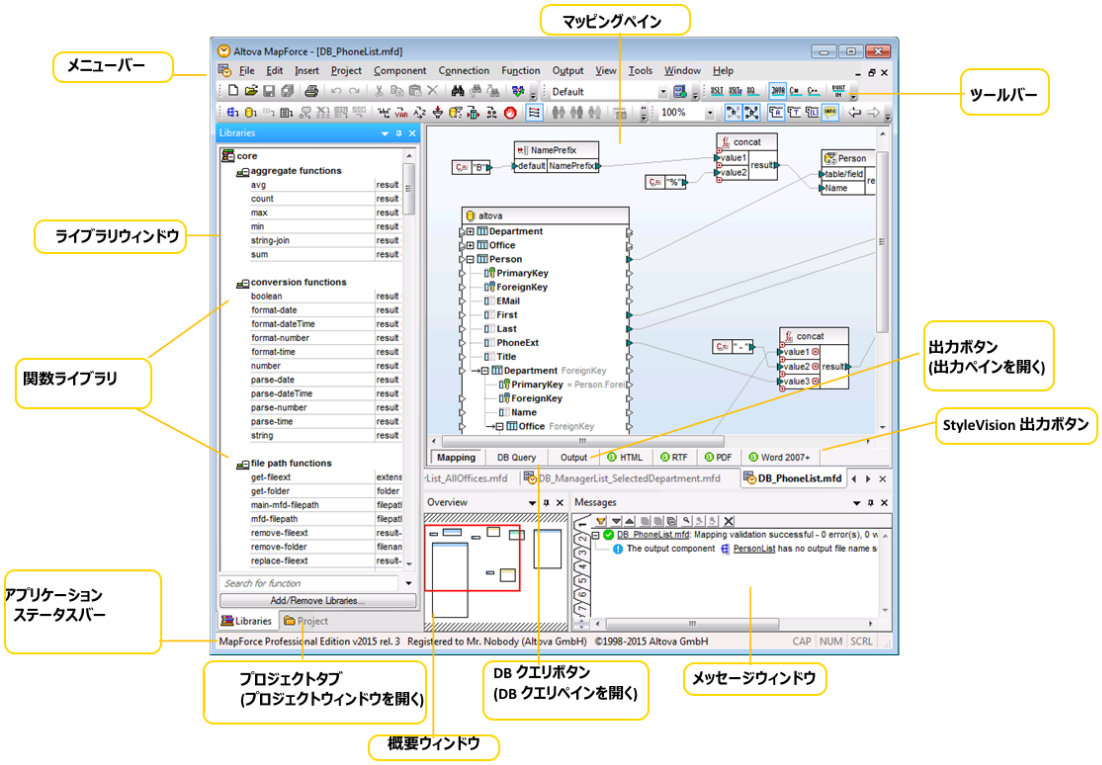
ターゲットコンポーネント

ターゲットコンポーネントとMapForce はがデータを書き込むコンポーネントです。マッピング を実行すると、ターゲットコンポーネントは、外部プログラム内で更に処理するためにMapForce にファイル (または複数のファイル)の生成、または結果を文字列の値として出力するように命令します。ターゲットコンポーネントは、ソース コンポーネントの逆です。

2.3 ユーザーインターフェイスの概要

MapForce のグラフィカルインターフェイスは、統合された開発環境として整理されています。メインのインターフェイス設定は **「ツール| カスタマイズ」** メニューコマンドを使用して行います。

各ウィンドウの右上に表示される ▾ 📌 ✕ ボタンを使用してウィンドウの、表示、非表示、ピン、ドックを行います。ツールバーとウィンドウをデフォルトの状態に復元する場合は、メニューコマンド **「ツール| ツールバーとウィンドウの復元」** を使用します。



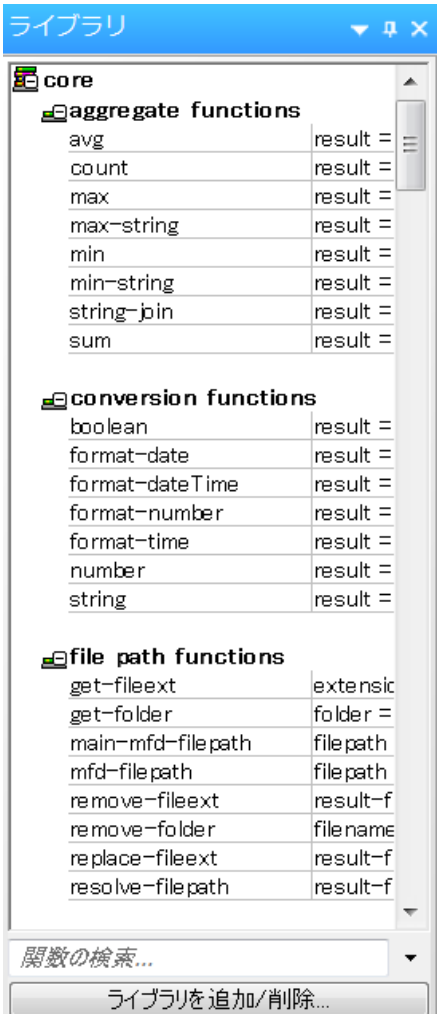
MapForce グラフィカルユーザーインターフェイス (MapForce Professional Edition)

メニューバーとツールバー

メニューコマンドは、メニューアイテムを表示します。各ツールバーは MapForce コマンドのボタンのグループを表します。希望する場所にハンドルをドラッグすることにより、ツールバーを移動することができます。

ライブラリウィンドウ

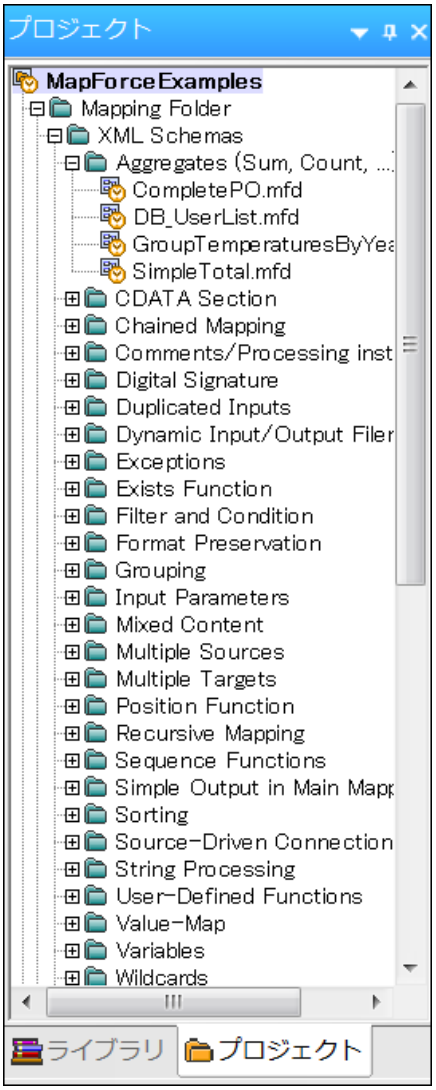
ライブラリウィンドウは、ライブラリごとに整理された MapForce 内蔵の関数をリストします。使用することのできる関数は選択する変換言語によりことなります。ユーザー定義関数を作成した場合、またはインポートした場合、これらの関数もライブラリウィンドウに表示されます。



名前または詳細による関数の検索は、**ライブラリ**ウィンドウの下のテキストボックスに検索の値を入力することにより行うことができます。現在アクティブなマッピング内で関数の全ての発生を検索するには、関数を右クリックして、コンテキストメニューから**全ての呼び出し**を選択します。また、**ライブラリ**ウィンドウから直接関数データ型と詳細を確認することができます。詳細に関しては、関数と作業を参照してください。

プロジェクトウィンドウ

MapForce 複数のドキュメントインターフェイスをサポートし、マッピングをマッピングプロジェクトとしてグループ化することができます。プロジェクトウィンドウはプロジェクトに追加されたすべてのファイルとフォルダーを表示します。プロジェクトファイルには、拡張子 *mfp (MapForce プロジェクト)があります。プロジェクト内でマッピングを検索するには、プロジェクト内をクリックして、**「CTRL + F」**を押します。詳細に関しては、マッピングプロジェクトとの作業を参照してください。



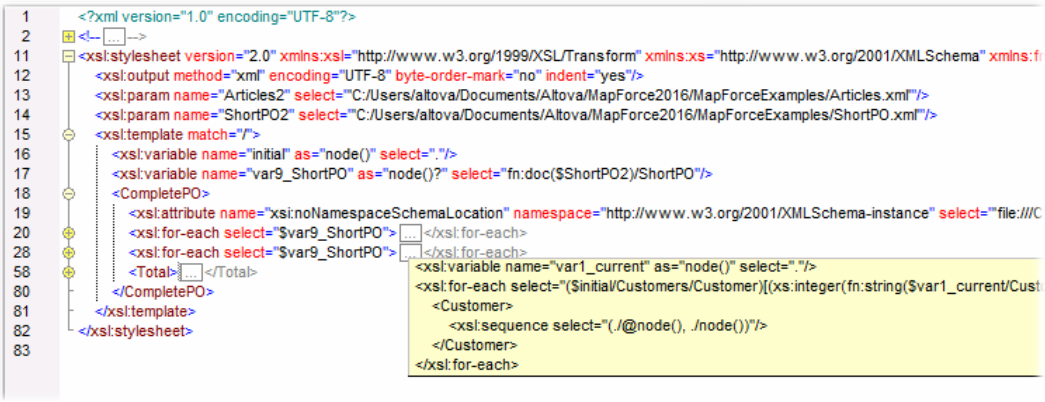
マッピング ペイン

マッピングペインはマッピングをデザインする作業エリアです。 **挿入** メニュー (コンポーネントをマッピングに追加を参照) からマッピングエリアに (ファイル、スキーマ、コンスタント、変数などの) マッピングコンポーネント追加することができます。 ライブラリウィンドウに表示されたマッピングペイン関数をドラッグすることもできます (関数と作業を参照)。

XSLT (XSLT2) ペイン

XSLT ペインは、 **「XSLT」** ボタンをクリックすると、マッピングから生成されたXSLT 1.0 変換コードを表示します。このペインはXSLT を変換言語として選択し、 **XSLT** タブ または **XSLT2** タブをクリックすると使用することができます。

このペインは、ラインの付番とコードの折りたたみ機能を提供します。コードの一部を展開または折りたたむには、ウィンドウの左側の「+」と「-」アイコンをクリックします。折りたたまれたコードは、省略記号と共に表示されます。折りたたまれたコードを展開することなくプレビューするには、マウスで省略記号をポイントすると、プレビューされるコードを表示したヒントが下に表示されるように開かれます。 プレビューされたテキストが大きくヒント内に表示できない場合、ヒントの後に追加表示記号が表示されます。



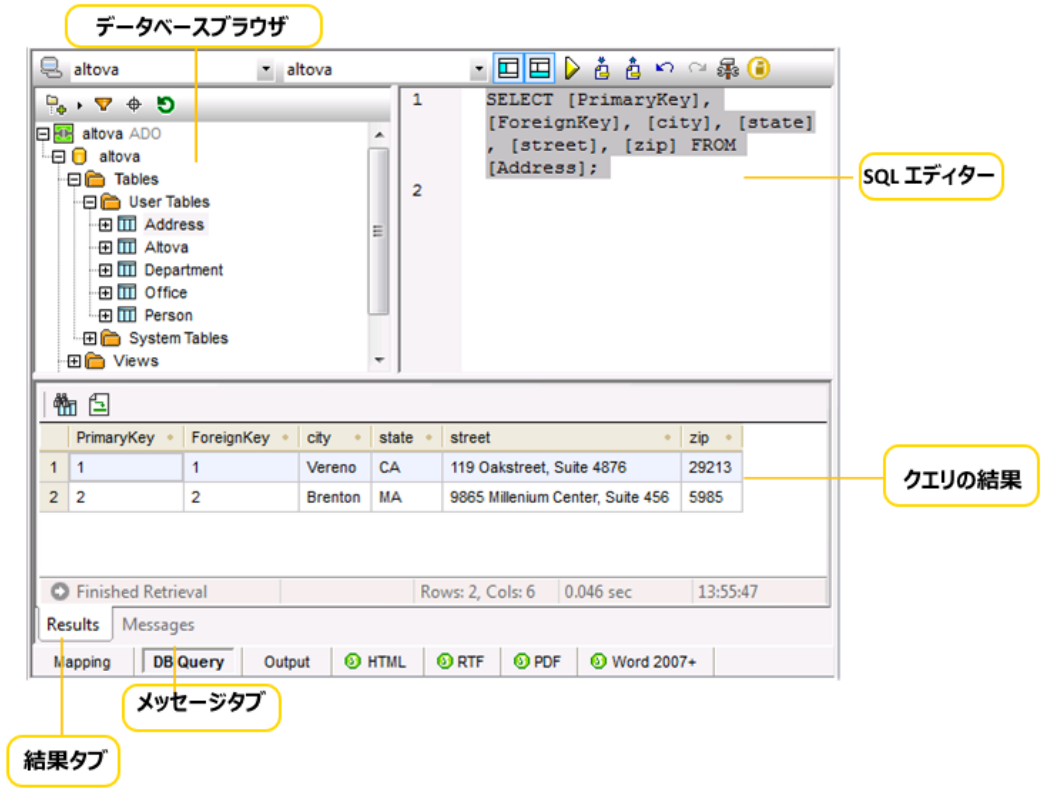
(インデント、行末マーカーを含む)表示設定を構成するには、ペインを右クリックして、コンテキストメニューから **「テキストビュー設定」**を選択します。または、 **「テキストビュー設定」**( ) ツールバーボタンをクリックします。

XQuery ペイン

XSLT ペインは、 **「XQuery 」**ボタンをクリックすると、マッピングから生成されたXQuery 変換コードを表示します。このペインはXQuery を変換言語として選択すると使用することができます。このペインもXSLT ペインと同様の仕組みを持つラインの付番とコードの折りたたみ機能を提供します (上を参照)。

DB クエリペイン

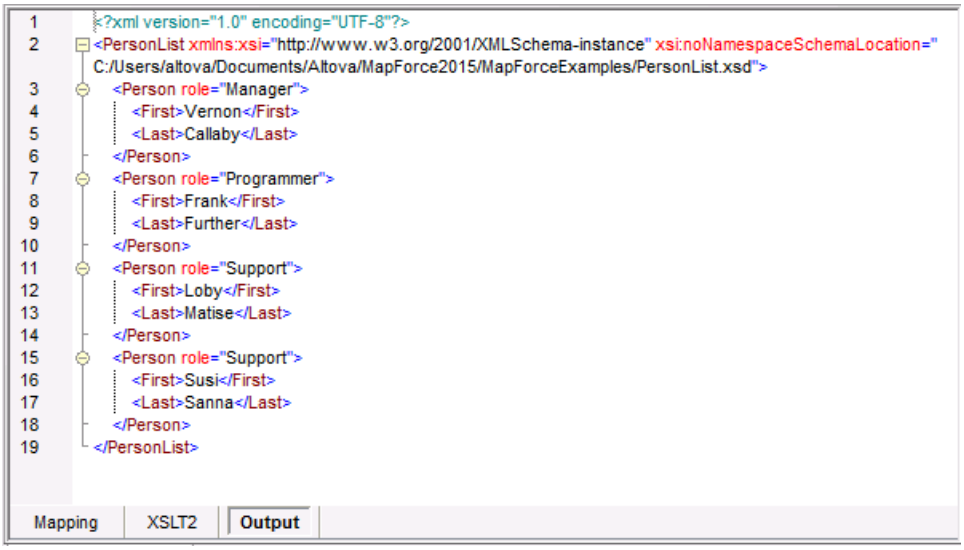
DB クエリペインでは全てのメジャーなデータベースに対して直接問い合わせを行うことができます。複数の異なるデータベースへのアクティブな接続と作業することができます。



更に詳しい情報に関しては、データベースの参照とクエリを参照してください。

出力ペイン

出力ペインは、**「出力」**ボタンをクリックするとマッピングの変換結果を表示します（例：XML ファイル）。マッピングが複数のファイルを生成する場合、それぞれの生成されたファイルを順番にナビゲートすることができます。



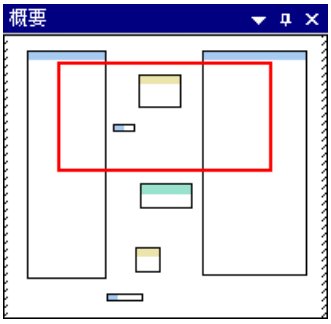
XSLT ペインと同様の仕組みを持つラインの付番とコードの折りたたみ機能を提供します（上を参照）。

StyleVision 出力ボタン

Altova StyleVision (http://www.altova.com/ja/stylevision.html) がインストールされている場合、StyleVision 出力ボタンによりマッピングの出力をでプレビューしHTML、RTF、PDF、とWord 2007+ フォーマットで保存することができます。これはStyleVision でデザインされた StyleVision Power StyleSheet (SPS) をMapForce 内のマッピングコンポーネントに割り当てることにより可能になります。

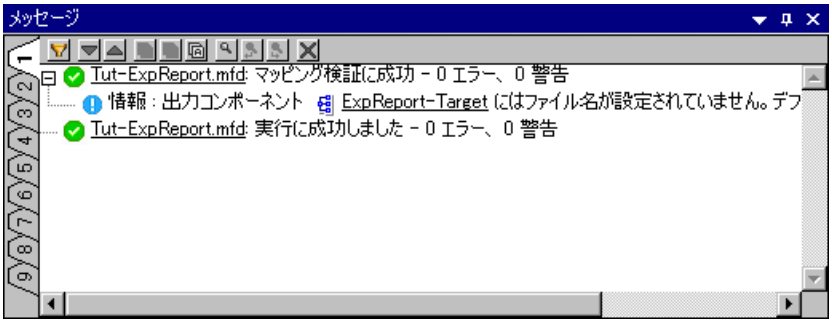
概要ウィンドウ

概要ウィンドウは、マッピングペインの概観図を表示します。マッピングのサイズが大きい場合、マッピングエリアの特定の場所に移動する場合に使用します。マッピングの特定の場所に移動する場合、赤い枠の長方形をクリックアンドドラッグします。



メッセージウィンドウ

メッセージウィンドウにはマッピング処理中 (出力のプレビューを参照)に行われる検証結果 (マッピングの検証を参照)の警告やエラーメッセージが表示されます。



情報、警告またはエラーメッセージをトリガーしたコンポーネントまたは構造のマッピングエリアをハイライトするには、メッセージウィンドウ内の下線のついたテキストをクリックします。

メッセージウィンドウ内のマッピング実行または検証オペレーションの結果は、以下のアイコンの1つと共に表示されます:

アイコン	説明
	オペレーションに成功しました。
	オペレーションは警告と共に終了しました。

アイコン	説明
	オペレーションに失敗しました。

メッセージウィンドウは次のメッセージを追加して表示する場合があります：情報メッセージ、警告、およびエラー

アイコン	説明
	情報メッセージを表示します。情報メッセージはマッピングの実行を停止しません。
	警告メッセージを表示します。警告はマッピングの実行を停止しません。例えば、必須の入力コネクタへの接続が作成されなかった場合、警告メッセージが生成されます。このような場合、有効な接続を持つコンポーネントのために出力が生成されます。
	エラーメッセージを表示します。エラーが発生すると、マッピングの実行に失敗し、出力は生成されません。XSLT または XQuery コードのプレビューの不可能です。

メッセージウィンドウ内の他のボタンにより以下のアクションを実行することができます：

アイコン	説明
	重要度（情報メッセージ、エラー、警告）別にメッセージをフィルターします。 **「すべて選択」**を選択して、全ての重要度を選択します（これはデフォルトの振る舞いです）。 **「チェックをすべて解除」**を選択して、フィルターからすべての重要度を解除します。この場合、全般的な実行または検証の状況に関するメッセージのみが表示されます。
	選択を次の行に移動させる。
	選択を前の行に移動させる。
	選択された行をクリップボードにコピーする。
	ネストされたラインを含む選択されたラインをクリップボードにコピーします。
	メッセージウィンドウの全てのコンテンツをクリップボードにコピーします。
	メッセージウィンドウ内で特定のテキストを検索します。オプションで、特定の用語を検索する場合は、 **「単語の完全マッチ」**を選択します。単語の大文字と小文字を区別する場合 **「大文字と小文字を区別」**を選択します。
	特定のテキストを現在選択されているラインから最後まで検索します。
	特定のテキストを現在選択されているラインから最初まで検索します。
	メッセージウィンドウをクリアする。

複数のマッピングファイルと同時に作業する場合、個々のマッピングのための個別のタブ内で情報、警告またはエラーを表示する必要があることがあります。この場合、マッピングを実行または検証する前に、メッセージウィンドウの左横にある番号の付いたタブをクリックします。

アプリケーションステータスバー

アプリケーションウィンドウの下部にあるアプリケーションステータスバーには、アプリケーションに関する情報が表示されます。ツールバーアイコンに対してマウスポインターを重ねると、ツールチップ情報がこのステータスバーに表示されます。アプリケーションステータスバーとデータベースクエリステータスバーは別のものです。64ビットバージョンの MapForce を使用している場合、アプリケーション名の後に (x64) という文字列が加えられます。32ビットバージョンでこのような

表示は行われません。

2.4 規約

サンプルファイル

このドキュメントで参照される(.mfd 拡張子 と他の伴うインスタンスファイルを持つファイル)データマッピングデザインファイルの多くは次のフォルダーで検索することができます:

- (My) Documents\Altova\MapForce2017\MapForce Examples
- (My) Documents\Altova\MapForce2017\MapForce Examples\Tutorials

(マイ)ドキュメント フォルダーの場所はWindows のバージョンにより異なります

Windows 10 Windows 8 Windows 7 Windows Vista Windows Server 2008 Windows Server 2012	`C:\Users\<username>\Documents`
Windows XP Windows Server 2003	`C:\Documents and Settings\<username>\My Documents`

MapForce に伴うサンプルマッピングとインスタンスファイルは動作の多くのアスペクトを紹介し、MapForce を使用するに当たり色々と試すことが奨励されます。オリジナルのサンプルへ直接変更を加えることを希望しない場合は、変更前にバックアップを作成することが奨励されます。

グラフィカルユーザーインターフェイス

このドキュメントに伴うイメージ (スクリーンショット)の一部は、使用中のMapForce エディションでは、使用することのできないグラフィカルユーザーインターフェイスの要素を含む場合があります。関連したコンテキストで、イメージは通常ソースマッピングデザイン (*.mfd) ファイルの名前、画像が作成されたMapForce のエディションを含みます。

Chapter 3

チュートリアル

3 チュートリアル

MapForce チュートリアルは、MapForce のデータの基本変換機能短い時間でを理解することをお手伝いします。チュートリアルの目的は全てのMapForce 機能を説明することではありませんが、MapForce の基本を順序を追って説明します。ですから、チュートリアルを順番どおりにフォローすることが奨励されます。コンセプトは順を追って複雑になるため、各コンセプトを次のコンセプトに移動する前に理解することが重要です。XML およびXML スキーマに対する基本的な知識はとても有利です。

XML を新しいスキーマに変換

このチュートリアルは、コードを書く必要なくXSLT 2.0 言語を使用してXML 構造から他にデータを変換する方法について説明します。また、 MapForce シーケンスとアイテム、マッピング接続作成、関数の使用方法、マッピングの検証およびプレビュー、と出力結果をディスクに保存する方法などについて学ぶことができます。

複数のソースから1つのターゲットにマップ

このチュートリアルは、異なるスキーマを持つ2つのXML ファイルからのデータの読み込み方法や単一のターゲットXML ファイルへのマージの方法を説明します。また、各マッピングコンポーネントの名前とインスタンスファイルの変更方法や入力の複製の方法などについて学ぶことができます。

複数のターゲットスキーマとの作業

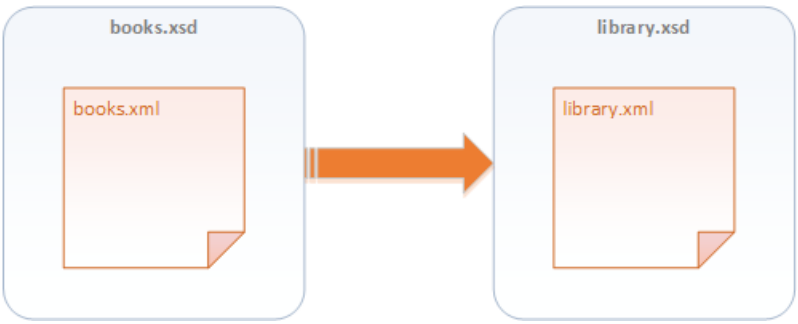
このチュートリアルは、2つ以上のターゲット出力を作成する更に複雑なマッピングとの作業方法について説明します。具体的な方法は、同じマッピング内でブックレコードのリストを保管するファイル、最初のXML ファイル内のブックのサブセットのみを含む他のXML ファイルを生成し、発行年によりフィルターします。データのフィルタリングをサポートするには、 **「フィルター」**コンポーネント、関数、数値定数を使用します。

ファイルを動的に処理と生成する

このチュートリアルは、同じフォルダーに存在する複数のXML インスタンスファイルからデータを読み込み複数のXML ファイルに書き込む方法を説明します。また、XML とスキーマ宣言 の削除および、関数を使用して、文字列を連結しファイル拡張子を抽出する方法についても学ぶことができます。

3.1 XML を新しいスキーマに変換

このチュートリアルは、MapForce 開発環境の基本を学びつつ、2つの XML ファイル間のデータの変換方法について説明します。両方の XML ファイルはブックのリストを保管しますが、その要素は、異なる方法により名前を付けられ整理されます。 (すなわち、2つのファイルは異なるスキーマを持ちます)。



データ変換の抽象的なモデル

下にリストされるコードは、データソースとして使用されるファイルからのサンプルのデータを表示しています (簡略化するために、XML と名前空間宣言は省略されています)。

```
<books>
   <book id="1">
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <category>Fiction</category>
      <year>1876</year>
   </book>
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
</books>
```

books.xml

ターゲット (デスティネーション) ファイルではデータは以下のように表示されます:

```
<library>
   <last_updated>2015-06-02T16:26:55+02:00</last_updated>
   <publication>
      <id>1</id>
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
   <publication>
      <id>2</id>
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <genre>Fiction</genre>
```

```
      <publish_year>1912</publish_year>
   </publication>
</library>
```

library.xml

ソースとターゲットXML 内の一部の要素の名前は、同じではありません。このセクションの目的は、ソースファイル (<author>、<title>、<category>、<year>) 内の同等の要素から、ターゲットファイルの<author>、<title>、<genre> と<publish_year> 要素を表示することです。ソースXML ファイル内の属性 id は、ターゲットXML ファイル内の<id> 要素にマップされる必要があります。最後に、ターゲットファイルの<last_updated> 要素をファイルが最後に更新された日付と時刻として表示するように設定する必要があります。

必要とされるデータ変換を行うには、以下のステップを行います。

ステップ 1：XSLT2 を変換言語として選択する

これを以下の方法で行うことができます：

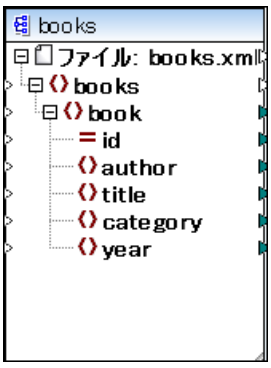
- **「XSLT2」**（ ）ツールバーボタンをクリックします。
- **「出力」** メニューから **「XSLT 2.0」** をクリックします。

ステップ 2：ソース XML ファイルをマッピングに追加する

このマッピングのためのソースXML ファイルは、は、以下のパスで見つけることができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\books.xml。** ファイルを以下の方法でマッピングに追加することができます：

- **「XML スキーマ /ファイルの挿入」**（ ）ツールバーボタンをクリックします。
- **「挿入」** メニューから **「XML スキーマ/ファイル」** をクリックします。
- XML ファイルをウィンドウエクスプローラーからマッピングエリアにドラッグします。

ファイルはマッピングエリアに追加され、構造を一見することができます。MapForce では、この構造はマッピングコンポーネントとして知られており、または単純にコンポーネント. コンポーネント内の要素を、折りたたむ (⊟) と展開するアイコン (⊞) をクリックすることにより、または数字キーパッドの **「+」** と **「-」** キーを押すことによりすることにより拡張することができます。



マッピング コンポーネント

マッピングペイン内でコンポーネントを移動するには、コンポーネントヘッダーをクリックして、マウスを新しい場所にドラッグします。コンポーネントのサイズを調整するには、コンポーネントの角をドラッグします ⌟。角をダブルクリックするとMapForce は自動的にサイズを調整します。

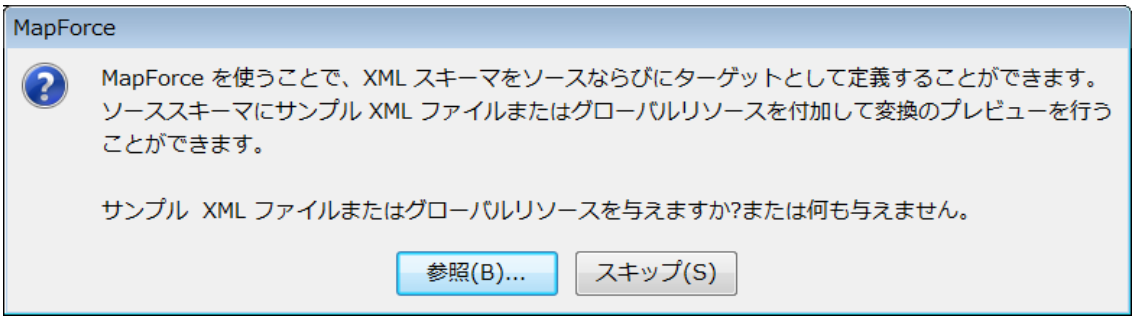
トップレベルノード は、ファイル名を表します。このサンプルの場合、は、XML インスタンスファイルの名前を表示します。構成内の、XML 要素は アイコンにより表示されており、XML 属性は アイコンにより表示されています。

コンポーネントの両側に表示されている小さな三角形は、(左側にある場合)データ入力を、(右側にある場合)データ出力を表します。MapForce では、それぞれ入力コネクタ、出力コネクタと呼ばれます。

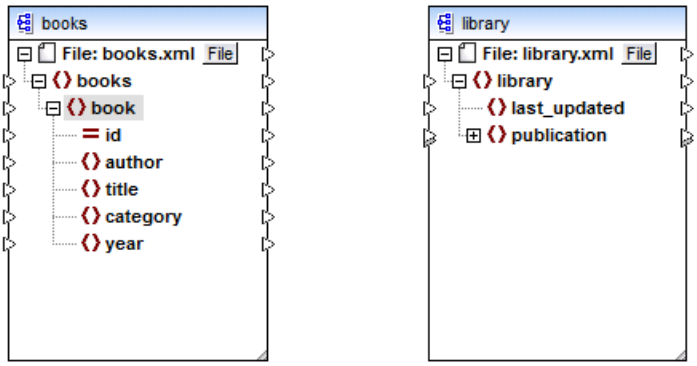
ステップ 3:ターゲット XML スキーマをマッピングに追加する

ターゲット XML を生成するには、既存の XML スキーマファイルを使用します。実生活のシナリオでは、このファイルは第三者パーティから提供される場合があり、または XMLSpy などのツールを使用してユーザーが作成することができます。XML データのためのスキーマファイルが存在しない場合、MapForce は伴うスキーマまたはスキーマレファレンスを持たない XML ファイルをマッピングに追加すると、スキーマを生成するようにプロンプトします。

このサンプルでは、以下で見つけることのできる既存のスキーマファイルを使用します:**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\library.xsd。** マッピングに追加するには、ソース XML ファイルと同様のステップを踏んでください。(**「XML スキーマ/ファイルの挿入」**( )ツールバーボタンをクリックします)。MapForce にインスタンスファイルを提供するようにプロンプトされると、**「スキップ」**をクリックします。



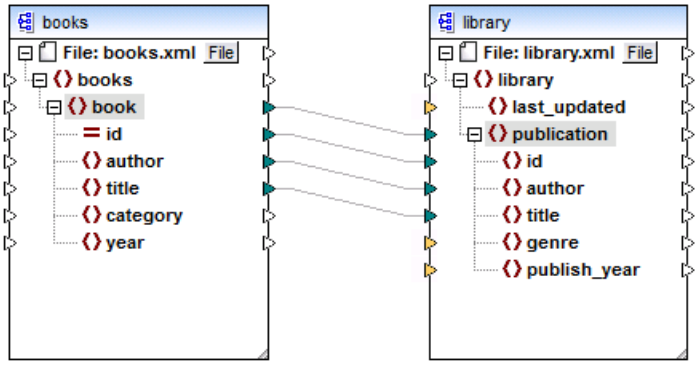
この時点では、マッピングデザインは、以下のようになります:



ステップ 4:接続の作成

ソース XML ファイルのそれぞれの <book> 内で、ターゲット XML ファイル内に新しい <publication> を作成します。 ですから、ソースコンポーネント内の <book> 要素とターゲットコンポーネント内の <publication> 要素を接続します。マッピング接続を作成するには、 <book> 要素の右の出力コネクタ(小さな三角形)をクリックし、ターゲット内の <publication> 要素の入力コネクタにドラッグします。

これを行うと、MapForce は自動的に、ターゲットファイル内に同じ名前を持つソースファイル内の<book> の子の全ての要素を接続します。ですから、4つの接続が同時に作成されます。この振る舞いは「子要素の自動接続」と呼ばれ、必要に応じて無効化またはカスタム化することができます。



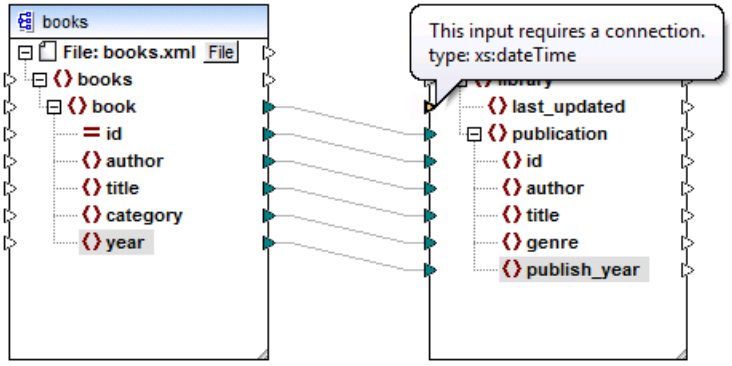
「マッチする子を自動的に接続する」の振る舞いを以下の方法で有効化または無効化することができます：

- **「子の自動接続の切り替え」**（ ）ツールバーボタンをクリックします。
- **「接続」**メニューから**「マッチする子を自動的に接続する」**をクリックします。

ターゲットコンポーネントの入力コネクタの一部が、MapForce により、必須アイテムを表示する、オレンジ色にハイライトされていることにご注意してください。XML ファイルの妥当性を確認するために、必須アイテムの値を以下のように提供してください：

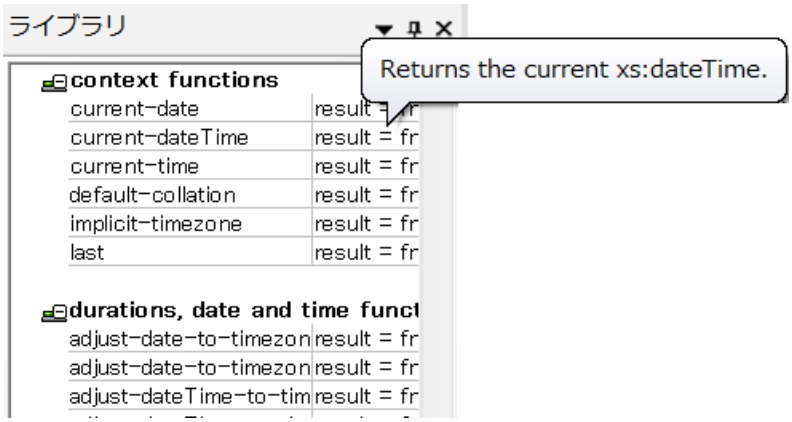
- ターゲット内の<genre> 要素とソース内の<category> 要素を接続する。
- ターゲット内の<publish_year> 要素とソース内の<year> 要素を接続する。

最後に、<last_updated> 要素への値を提供する必要があります。入力コネクタにマウスをポイントすると、要素は xs:dateTime 型であることが表示されます。ヒントを表示するには、**「ヒントの表示」**（ ）ツールバーボタンが有効化されている必要があることにご注意してください。



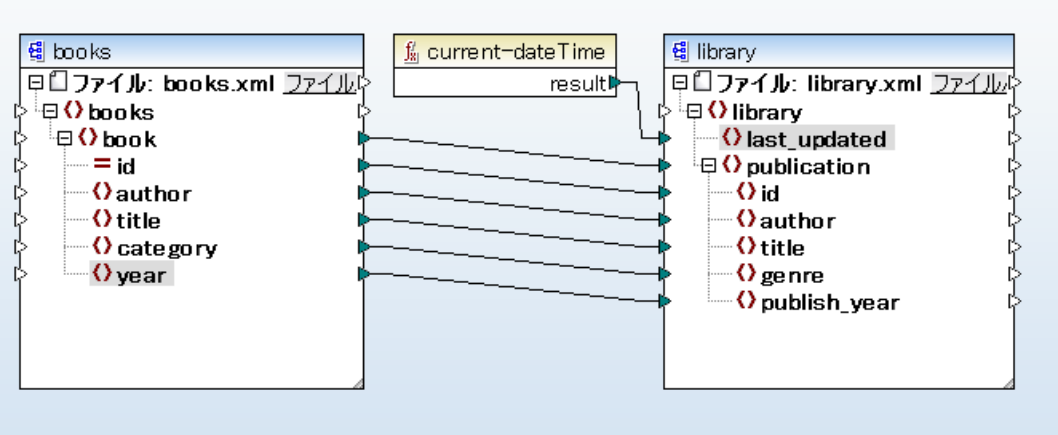
「データ型の表示」（ ）ツールバーボタンをクリックすることにより、常に各アイテムのデータ型を表示することができます。

現在の日時（すなわち、`xs:dateTime` の値）を関 XSLT 数の日付と時刻で取得することができます。マッピングに対するXSLT 関数を検索するには、ライブラリウィンドウの下の部分にあるテキストボックスに「date」を入力します。



上に表示されるように、マウスを関数の結果の部分にかざすと、詳細が表示されます。ヒントを表示するには、**「ヒントの表示」**（ ）ツールバーボタンが有効化されていることを確認してください。

マッピングに関数を追加するには、関数をマッピングペインにドラッグし、`<last_updated>` 要素の出力を入力に接続します。



(**books.xsd** スキーマを持つ)**books.xml** インスタンスファイルのデータから(**library.xsd** スキーマを持つ) **library.xml** ファイルへデータを変換するMapForce マッピングデザイン(または「マッピング」)を作成することに成功しました。各コンポーネントのヘッダーをダブルクリックすると、コンポーネント設定ダイアログボックス内のこれらと他の設定を確認することができます。



ソースのためのコンポーネント設定



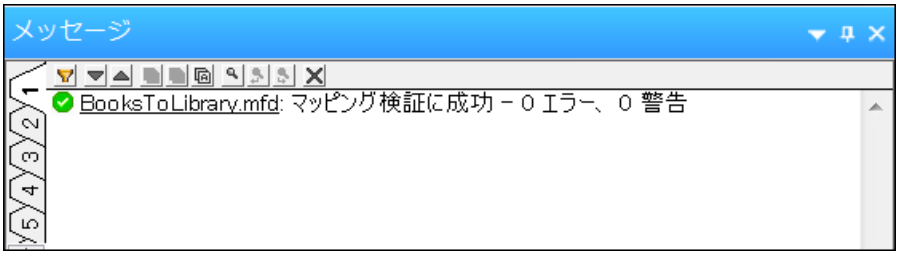
ターゲットのコンポーネント設定

ステップ 5：マッピングを検証し保存する

マッピングの検証は、マッピングを実行する前に、マッピングエラーの可能性を確認し修正するための任意のステップです。マッピングが妥当であるか確認するため、以下を行います：

- **「ファイル」**メニューから**「マッピングの検証」**をクリックします。
- **検証」**（ ）ツールバーボタンをクリックします。

「メッセージ」ウィンドウは、検証の結果を表示します：



メッセージウィンドウ

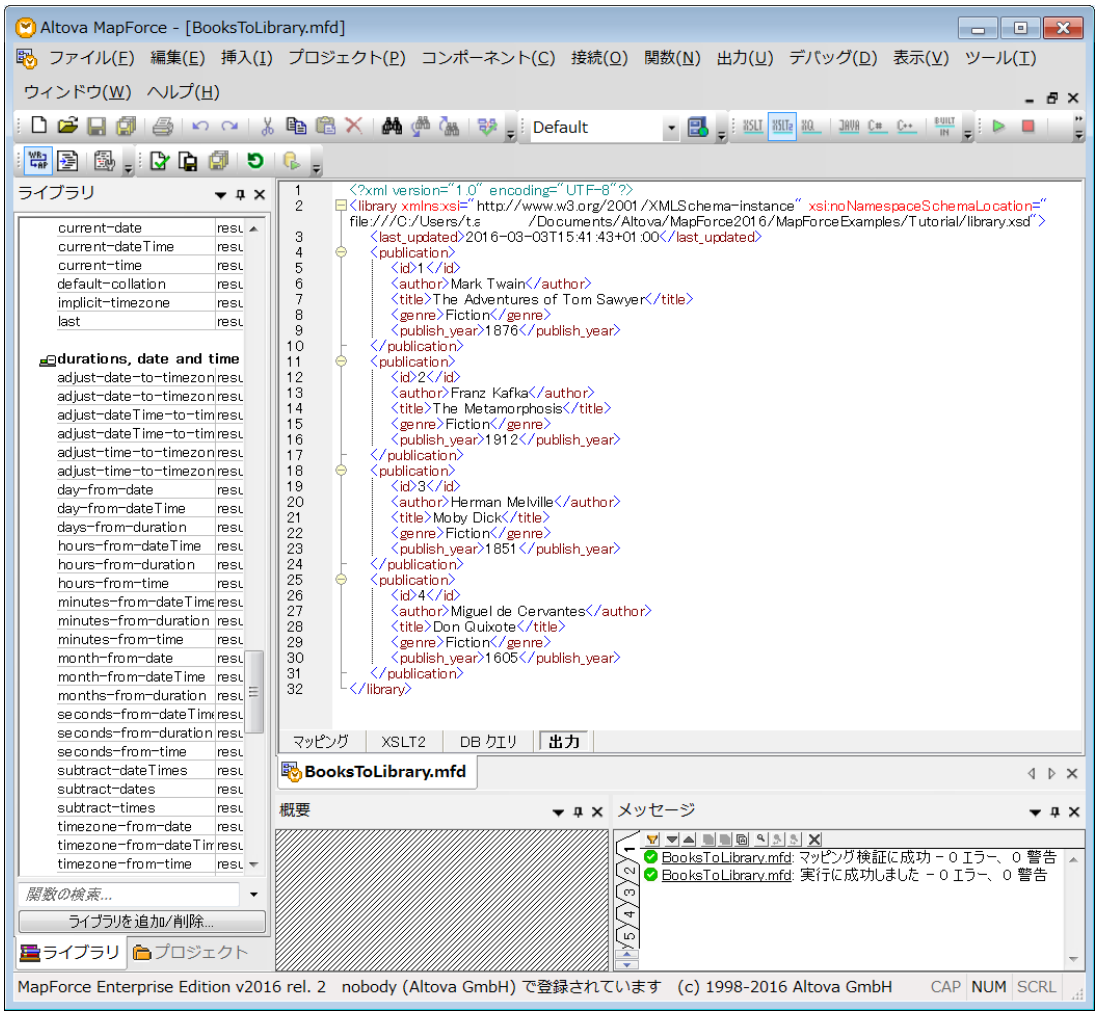
この時点で、ファイルにマッピングを保存します。マッピングを保存するには、以下を行います：

- **「ファイル」**メニューから**「保存」**をクリックします。
- **保存」**（ ）ツールバーボタンをクリックします。

このチュートリアルで作成されたマッピングは以下で検索することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\\BooksToLibrary.mfd**。ですから、これから先、自分で作成したマッピングを継続して使用するか、または**BooksToLibrary.mfd** ファイルを使用することができます。

ステップ 6：マッピングの結果のプレビュー

マッピングの結果の結果を直接でMapForce でプレビューすることができます。これを行うには、マッピングペインの下にある、**「出力」**ボタンをクリックします。MapForce は、変換を実行し、マッピングの結果を**「出力」**ペインに表示します。



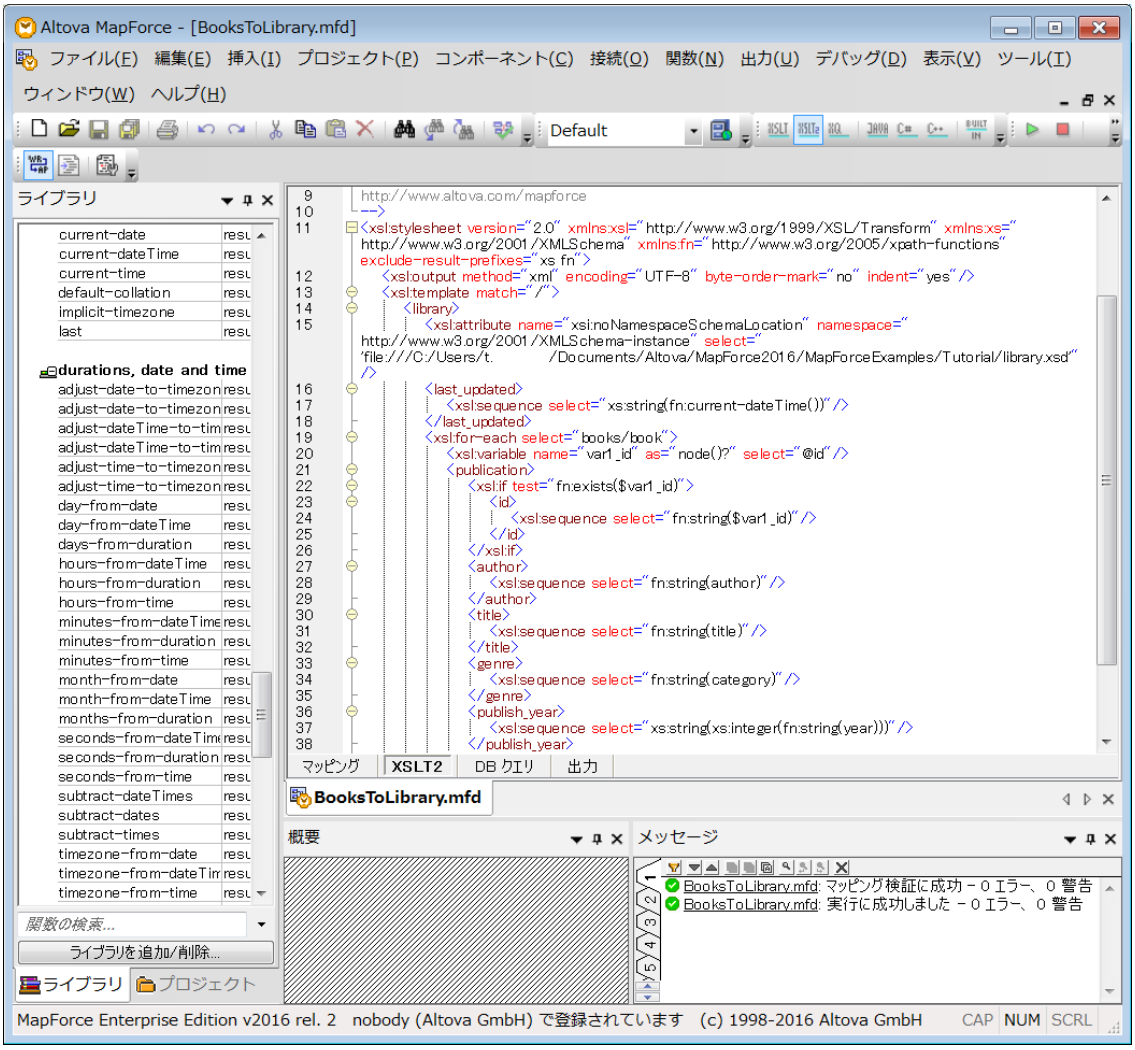
出力ペイン

MapForce 内で変換の結果を確認することができます。

デフォルトでは、**「出力」**ペイン内にプレビューのために表示されたファイルはディスクに書き込まれません。その代わり、MapForce は一時ファイルを作成します。**「出力」**ペインに表示されたファイルをディスクに保存するには、メニューコマンド **「出力 | 出力ファイルの保存」**を選択するか、または **「生成された出力を保存」**( ) ツールバーボタンをクリックします。

MapForce が一時ファイルではなく最終のファイルに直接的に出力を書き込むように構成するには、**「ツール | オプション | 全般」**に移動し、**「最終出力ファイルにファイルを書き込む」**チェックボックスを選択します。チュートリアルを実行中は、故意にではなくもとのチュートリアルファイルを上書きする可能性があるため、このオプションを有効化することは奨励されません。

変換を行う生成された XSLT コードもプレビューすることができます。コードをプレビューするには、マッピングペインの下にある、**「SLT2」**ボタンをクリックします。



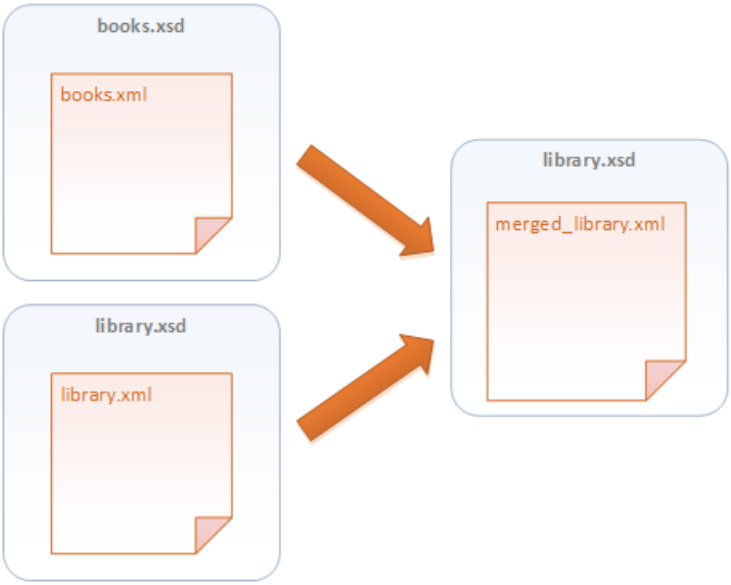
XSLT2 ペイン

XSLT2 コードを生成しファイルに保存するには、メニューアイテム **「ファイル | コード生成 | XSLT 2.0」** を選択します。プロンプトされると、生成されたコードが保存されるフォルダーを選択してください。コード生成が完了すると、デスティネーションファイルは、以下の2つのファイルを含みます：

1. ターゲットスキーマにちなんで名づけられたXSLT 変換ファイル (このサンプルでは、**MappingMaptolibrary.xslt**)。
2. **DoTransform.bat** ファイル。**DoTransform.bat** ファイルによりRaptorXML Server 内でのXSLT 変換の実行を有効化することができます。(詳細に関しては、http://www.altova.com/ja/raptorxml.html を参照してください)。

3.2 複数のソースから1つのターゲットにマップ

前のチュートリアルでは、ソースファイル (**books.xml**) からデータをターゲットファイル (**library.xml**) に変換しました。ターゲットファイル (**library.xml**) はマッピングの実行前は存在しておらず、マッピングの変換により生成されました。**library.xml** ファイル内にすでにデータが存在すると仮定し、このデータを**books.xml** から変換されたデータとマージするとします。このチュートリアルの目的は、**merged_library.xml** という名前のファイルを生成するマッピングをデザインすることです。生成されたファイルは、以下の2つのソースからのデータを含みます: **books.xml** ファイルと**library.xml** ファイル。ソースとして使用されるファイル (**books.xml** と**library.xml**) は異なるスキーマを持つことに注意してください。ソースファイルが同じスキーマを持つ場合、異なるアプローチを使用してデータをマージすることができます (ファイルを動的に処理と生成するを参照)。



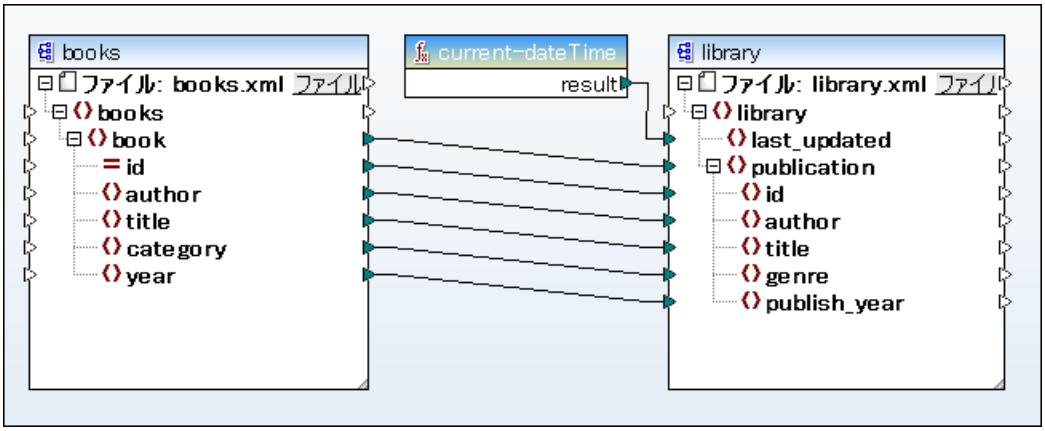
データ変換の抽象的なモデル

チュートリアルの目的を達成するためには、次のステップを踏みます。

ステップ 1: マッピングデザインファイルの準備

このチュートリアルは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーからの**BooksToLibrary.mfd** マッピングを開始点として使用します。このマッピングはXML を新しいスキーマに変換するチュートリアル内で作成済みです。チュートリアルを開始するには、MapForce で**BooksToLibrary.mfd** ファイルを開き、新しい名前で保存します。

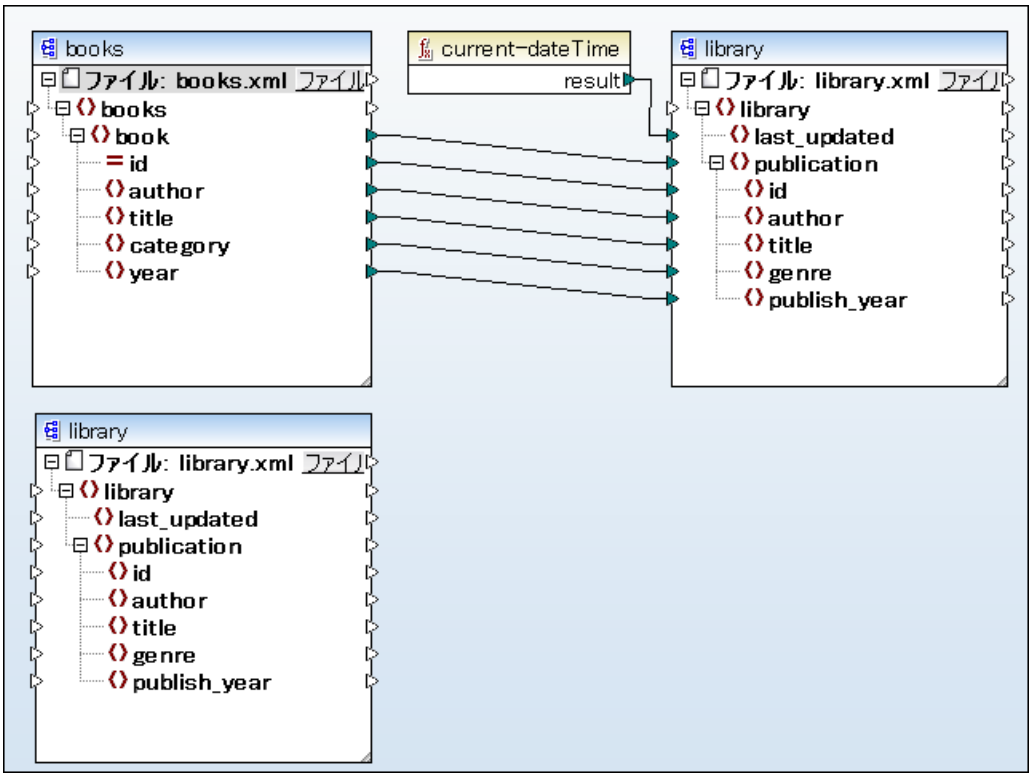
> 複数のファイルを参照するため、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内に新しいマッピングが保存されることを確認してください。



BooksToLibrary.mfd (MapForce Basic Edition)

ステップ 2 : 2番目のソースコンポーネントの作成

最初に、ターゲットコンポーネントを選択して、（**Ctrl + C**」を押して）コピーし、同じマッピング内に（**Ctrl + V**」を押して）貼り付けます。 新しいコンポーネントのヘッダーを**books** コンポーネントの下にドラッグします。



マッピングには2つのソースコンポーネントと:**books** と**library**、と 1つのターゲットコンポーネントがあります :**library**。

マッピングコンポーネントを4つの (左、右、上、下) 方向に移動することができます。しかしながら、ソースコンポーネントをターゲットコンポーネントの左に配置することにより、マッピングの作成を簡単にし、また、他のユーザーによる読み込みやすく理解しやすくなります。このドキュメントで説明されている全てのマッピングにもこの規則は適応されており、また、MapForce インストールに伴うサンプルマッピングファイルにも同様の規則が適用されています。

ステップ 3 : 入力 /出力ファイルの検証とセット

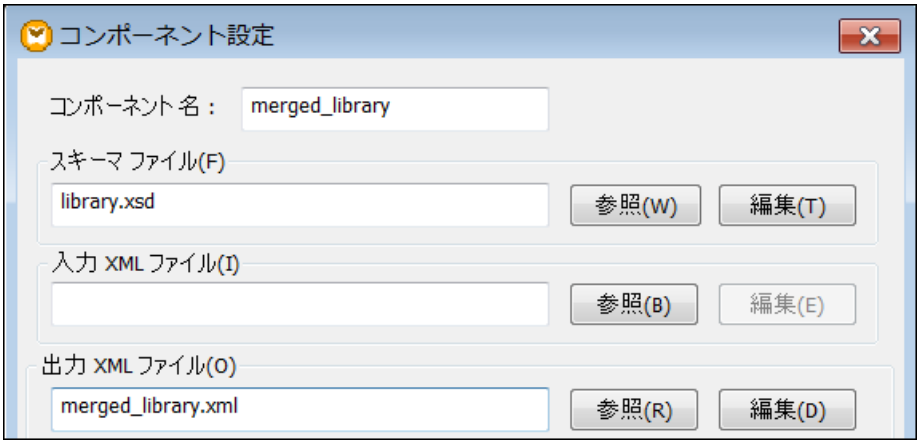
前のステップでは、新しいソースコンポーネントはターゲットコンポーネントからコピーし、貼り付けられましたので同じ設定を継承します。名前入力 /出力インスタンスファイルが正確に設定されるように、各コンポーネントのヘッダーをダブルクリックし、コンポーネント設定ダイアログボックス内で、下に表示されるように各コンポーネントの入力 /出力ファイルを検証し、名前を変更します。



*最初のソースのコンポーネント設定 (**books**)*



*2番目のソースのコンポーネント設定 (**library**)*

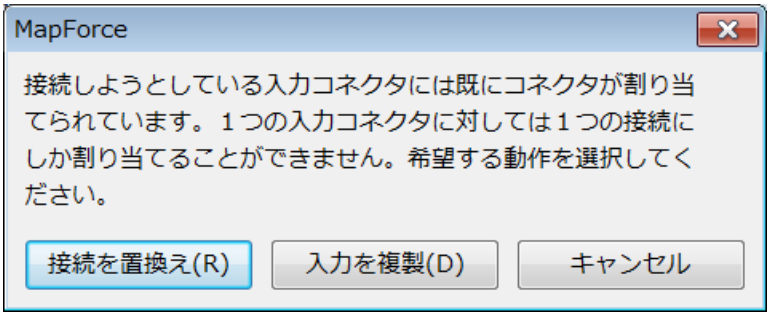


*ターゲットのコンポーネント設定 (**merged_library**)*

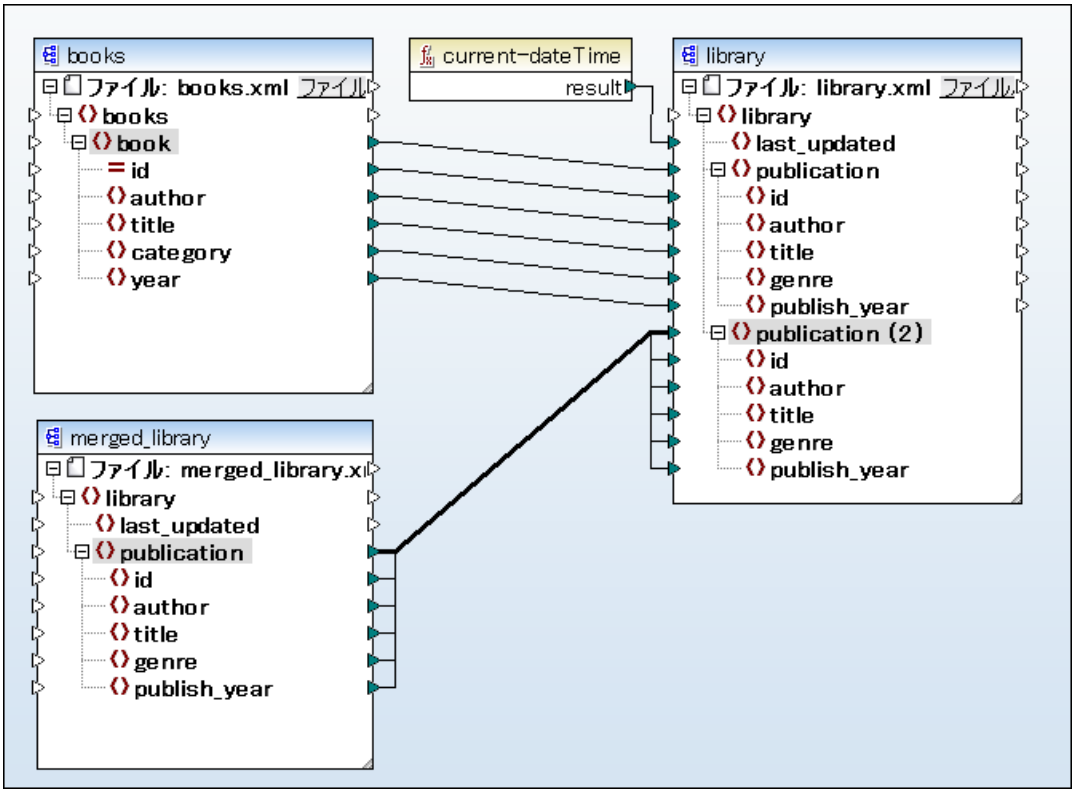
上に表示されるように、最初のソースコンポーネントは**books.xml** からデータを読み込みます。2番目のソースコンポーネントは**library.xml** からデータを読み込みます。2番目のソースコンポーネントは**merged_library.xml** というファイルのターゲットコンポーネント出力データを最後に読み込みます。

ステップ 4：接続の作成

MapForce に2番目のソースからターゲットにデータを書き込むように命令するには、ソース**library** 内コンポーネントの `publications` アイテムの出力コネクタ (小さな三角形) をクリックし、ターゲット**library** コンポーネント内のアイテムの入力コネクタにドラッグします。ターゲット入力コネクタには既に接続があり、次の通知メッセージが表示されます。



このチュートリアルでは、接続の置換えが目的ではなく、目的は2つのソースからのデータのマッピングです。ですから、**「入力の複製」**をクリックします。これを行うことにより、ターゲットコンポーネントに新しいソースからのデータを受け入れるように構成することができます。マッピングは、以下のようになります：



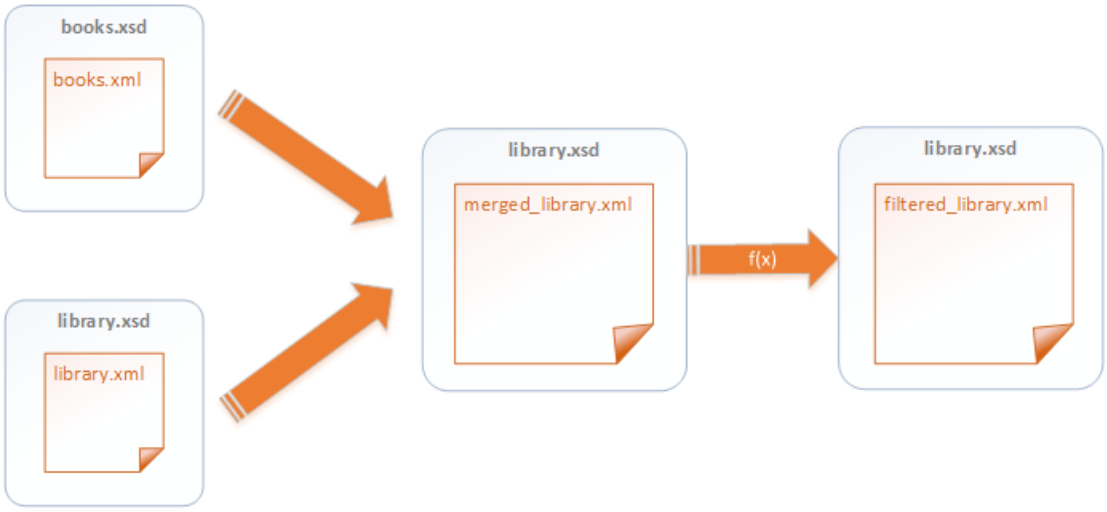
ターゲットコンポーネント内の`publication`アイテムが複製されました。新規`publication(2)`ノードはソース**library**コンポーネントからのデータを受け入れます。更に重要な点は、マッピング内でこのノードは`publication(2)`と表示されますが、結果XMLファイル内での名前は、想定された目的である`publication`となります。

マッピングペインの下の**「出力」**ボタンをクリックして、マッピングの結果を確認します。**library.xml**と**books.xml**ファイルのデータが、新規ファイル**merged_library.xml**にマージされたことを確認してください。

3.3 複数のターゲットスキーマとの作業

前のチュートリアルの、複数のソースから1つのターゲットにマップでは、複数のソーススキーマから単一のターゲットスキーマへのデータのマッピングについて作成方法について説明しました。2つのソースからのブックのレコードを保管する **merged_library.xml** と呼ばれるファイルを作成しました。他の部署からこのXML ファイルのサブセットを提供するように要請があったと仮定します。特に、1900年以降に出版された書籍を含むXML ファイルを提出するとします。

既存の**MultipleSourcesToOneTarget.mfd** マッピングを使用し、完全なXML ライブラリとフィルターされたライブラリ必要に応じて、を生成することができます。変更を加えることができます。



データ変換の抽象的なモデル

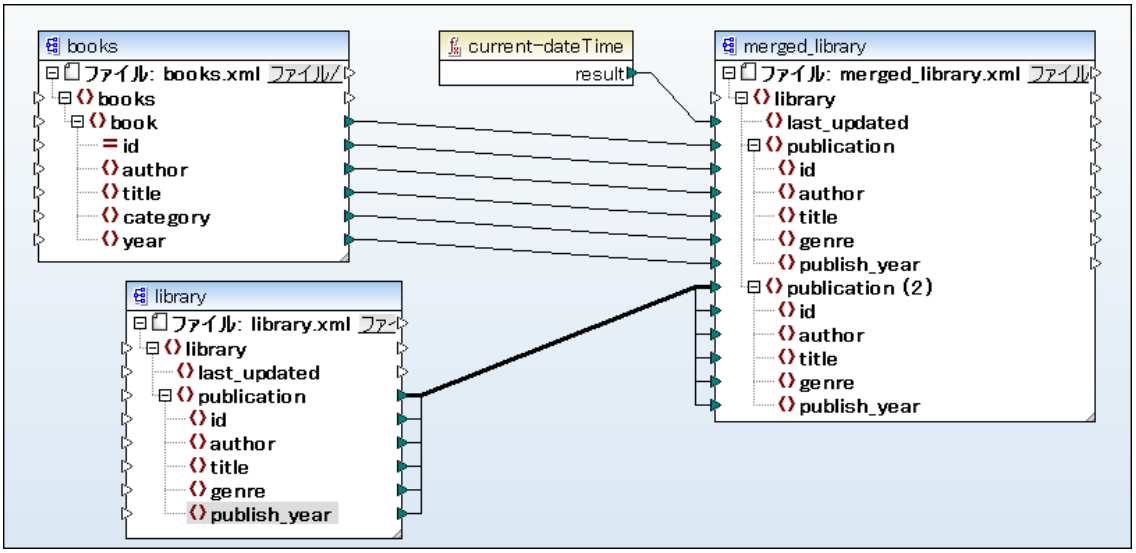
上の図では、データは2つの異なるスキーマ (**books.xsd** と**library.xsd**) が**merged_library.xml** と呼ばれる単一のXML ファイルにマージされます。第2に、データはフィルター関数を使用して変換され、**filtered_library.xml** と呼ばれるXML ファイルを作成する、次のコンポーネントにパスされます。「中間」コンポーネントは、データターゲットとソースとしての役割を果たします。MapForce では、このテクニックは「チェーンマッピング」と呼ばれ、このチュートリアルの主な内容です。ます。

このチュートリアルのゴールは、**merged_library.xml** と**filtered_library.xml** を必要に応じて生成することです。チュートリアルの目的を達成するためには、次のステップを踏みます。

ステップ 1：マッピングデザインファイルの準備

このチュートリアルは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーからの**MultipleSourcesToOneTarget.mfd** マッピングを開始点として使用します。このマッピングは複数のソースから1つのターゲットにマップチュートリアル内で作成済みです。チュートリアルを開始するには、MapForce で **MultipleSourcesToOneTarget.mfd** ファイルを開き、新しい名前で保存します。

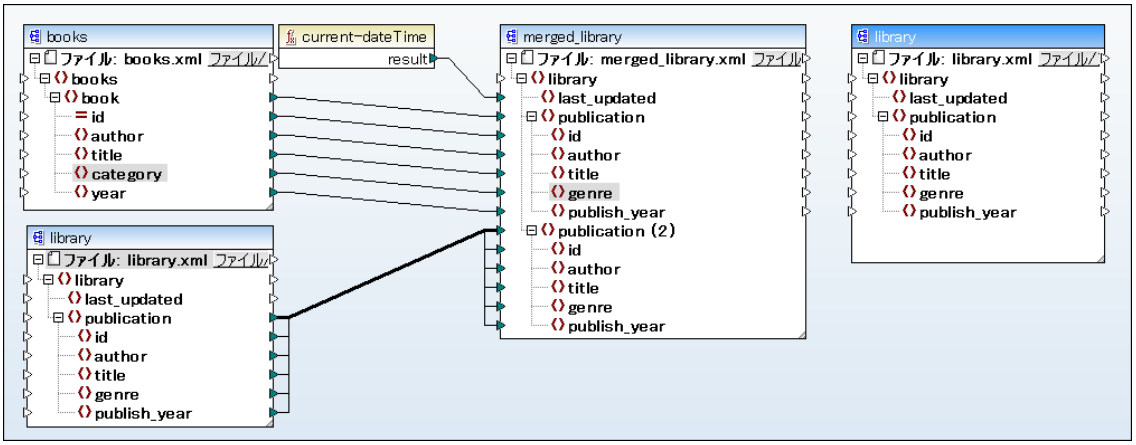
> 複数のファイルを参照するため、<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\ フォルダー内に新しいマッピングが保存されることを確認してください。



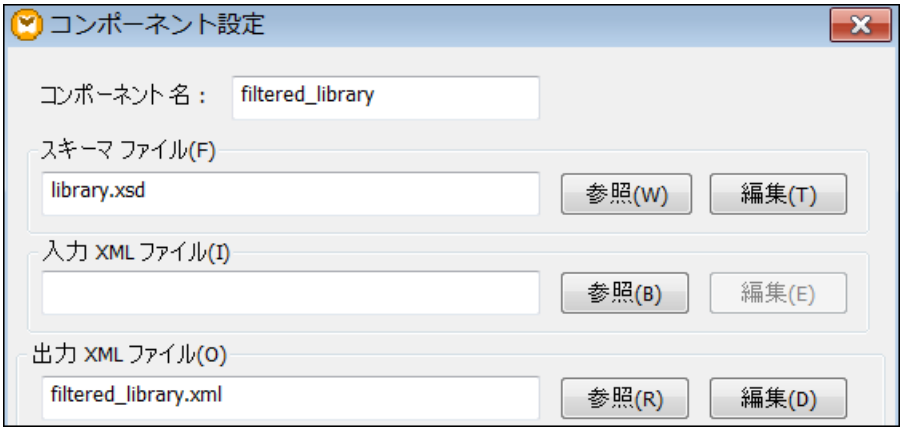
MultipleSourcesToOneTarget.mfd (MapForce Basic Edition)

ステップ 2：2番目のターゲットコンポーネントの追加と構成

2番目のターゲットコンポーネントを追加するには、**「XML スキーマ /ファイルの挿入」**（ ）ツールバーボタンをクリックし、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーにある **library.xsd** ファイルを開きます。サンプルインスタンスファイルを提供するようにプロンプトされた場合、**「スキップ」** をクリックします。マッピングは、以下のようになります：



上に表示されるように、マッピングには2つのソースコンポーネントと**:books** と**library**、と 2つのターゲットコンポーネントがあります。ターゲットコンポーネントを区別するために、2番目のターゲットコンポーネントを**filtered_library** に名前を変更し、生成されるXML ファイルの名前も変更します。これを行うには、右端にあるコンポーネントをダブルクリックして、コンポーネントの設定を次のように編集します：

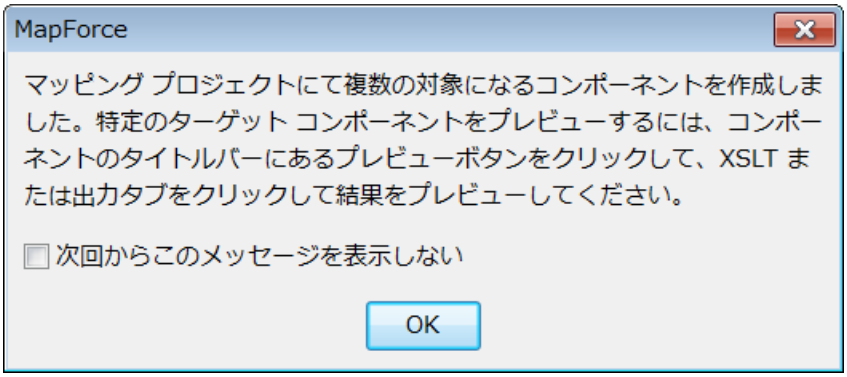


コンポーネントの新しい名前は**filtered_librar** となり、出力 XML ファイルには**filtered_library.xml** という名前が付けられます。

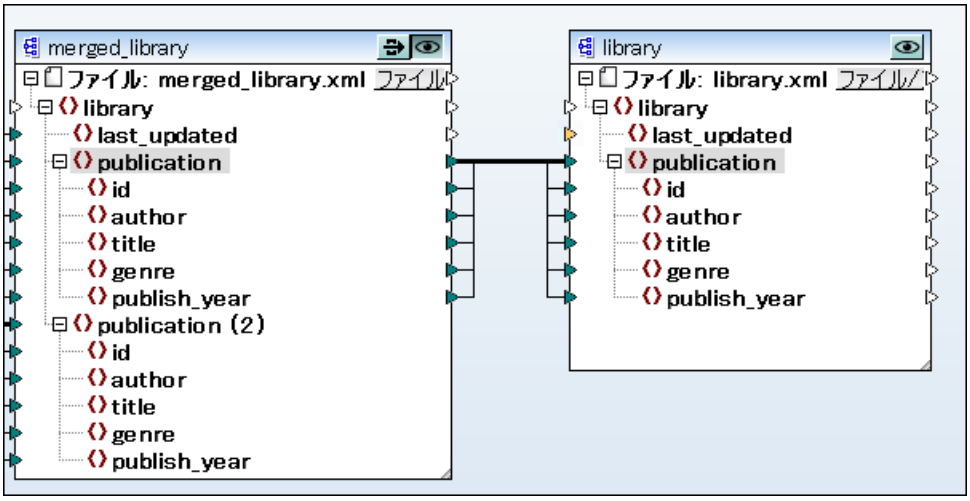


ステップ 3：接続の作成

merged_library 内のアイテム**publication** から**filtered_library** 内のアイテム**publication** への接続を作成します。これを行う際、通知メッセージが表示されます。

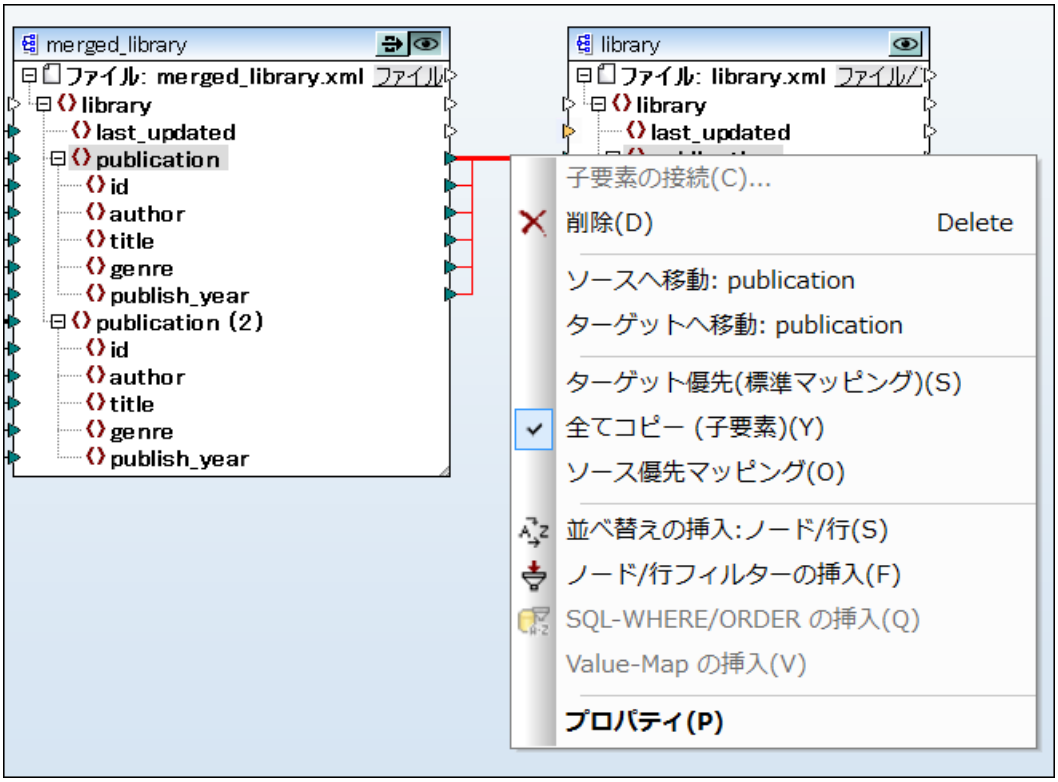


「OK」をクリックします。次の新規のボタンがターゲットコンポーネントの右上に表示されます：**「プレビュー」**（ ）と**「パススルー」**（ ）。次のステップでこれらのボタンは使用され、説明されます。

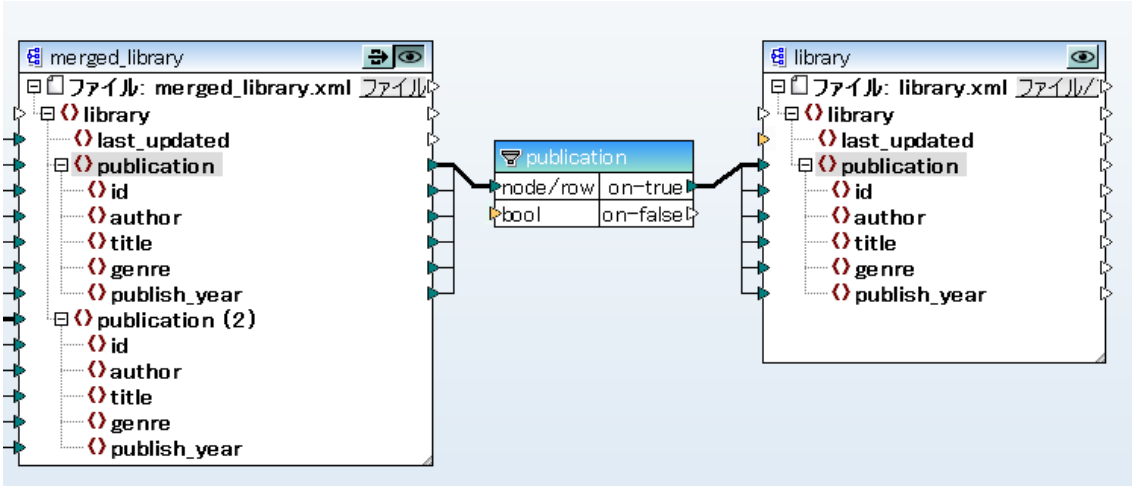


ステップ 4：データのフィルター

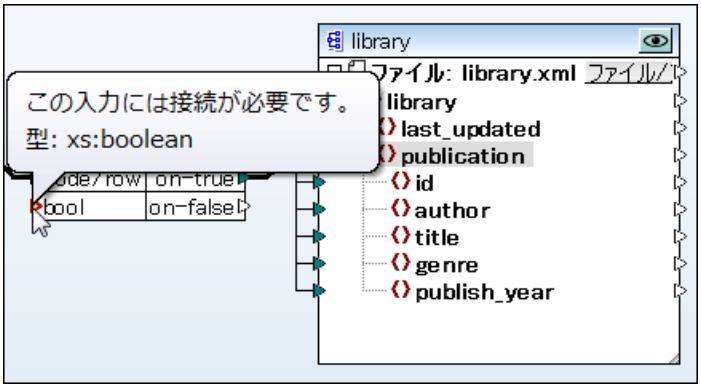
filtered_library に提供する前にデータをフィルターするには、**「フィルター」**コンポーネントを使用します。フィルターコンポーネントを追加するには、**merged_library** と**filtered_library**、間の接続を右クリックして、コンテキストメニューから、**「Node/row フィルターの挿入」**を選択します。



フィルターコンポーネントがマッピングに追加されました。



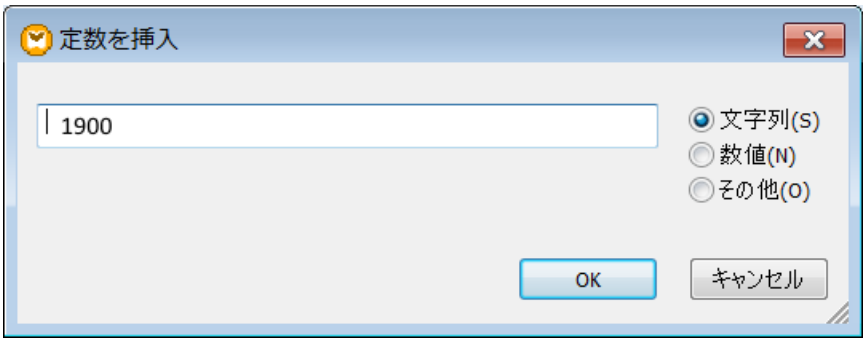
上に表示されるように、ブール入力コネクタは、入力が必要なことを示すオレンジ色にハイライトされます。コネクタ上にマウスをポイントすると、入力の型 `xs:boolean` が必要なことがわかります。ヒントを表示するには、**「ヒントの表示」**( )ツールバーボタンが有効化されている必要があることに注意してください。



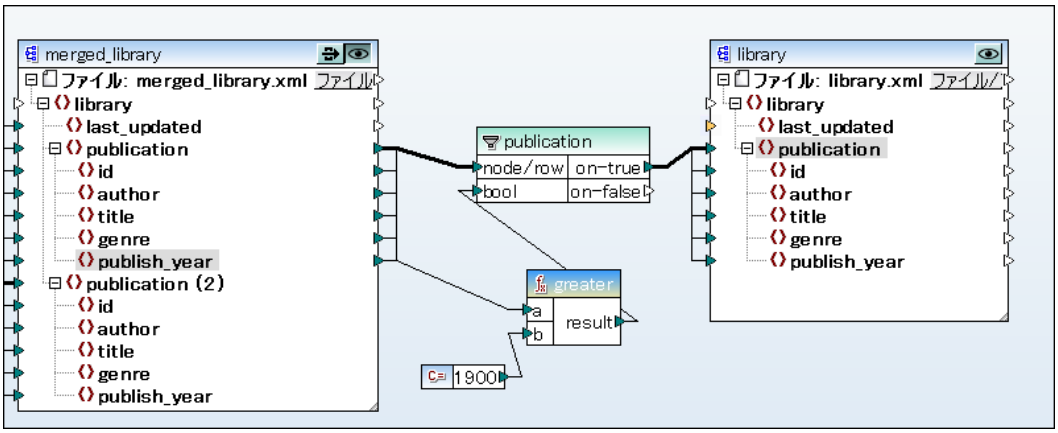
フィルターコンポーネントは`true` または`false` を返す条件を必要とします。ブール型の条件は、現在の**publication** シーケンスのデータがターゲットにコピーされると`true` を返します。条件が、`false` を返すと、データはコピーされません。

このチュートリアルで使用される条件は1900年以降に出版された書籍をフィルターします。条件を整えるために、以下を行います：

1. 1900 の値を持つ定数の数値の型を追加します。(**挿入**」メニューから**定数**」をクリックします)。**数値**」を型として選択します。



2. ライブラリウィンドウ内で、関数 **greater** を検索しマッピングペインにドラッグします。
3. **greater** 関数へと**greater** 関数から以下に表示されるように、マッピングを接続します。これを行うことにより、MapForce に以下を行うように命令します：「`publish_year` (出版年) が 1900以降の場合は、現在の`publication` ソースアイテムを`publication` ターゲットアイテム」にコピーします。



ステップ 5：各ターゲットコンポーネントの出力をプレビューし保存する

ターゲットコンポーネントの出力をプレビューし保存することができます。複数のターゲットコンポーネントが同じマッピングに存在する場合、 **「プレビュー」**（ ）ボタンをクリックすることにより、プレビューの対象を選択することができます。 **「プレビュー」**ボタンが押された状態（ ）の場合、特定のコンポーネントが現在プレビューのために有効化されていることを示します（そして、この特定のコンポーネントが「プレビュー」ペイン内に出力を生成します）。一度に1つのコンポーネントのみがプレビューのために有効化することができます。

ですから、**merged_library**（すなわち、「中間」）コンポーネントの出力を確認し保存する場合は、以下を行います：

1. **merged_library** コンポーネントの **「プレビュー」**ボタン（ ）をクリックします。
2. **「出力」**ボタンマッピングペインの下のをクリックします。
3. 出力をファイルに保存するには、 **「出力」** メニューから **「出力ファイルの保存」**をクリックします。

filtered_library コンポーネントの出力を表示して保存するには以下を行います：

1. **merged_lilbrary** コンポーネント上の **「パススルー」**ボタン（ ）をクリックします。
2. **filtered_library** コンポーネント上の **「プレビュー」**ボタン（ ）をクリックします。
3. マッピングペインの下の **「出力」** ボタンをクリックします。
4. 出力をファイルに保存するには、 **「出力」** メニューから **「出力ファイルを保存」**をクリックします。

「パススルー」（ ）ボタン? をクリックまたはクリックしないことにより、複数のターゲットコンポーネントを持つマッピングに大きな違いをもたらします。このボタンが押された状態（ ）である場合、MapForce は、データに中間コンポーネントをパススルーさせ、マッピング全体の残りをプレビューすることができます。

merged_library と**filtered_library** 間のマッピングの部分のみをプレビューするには、ボタン（ ）をリリースします。後者の場合、エラーが生成されます。中間コンポーネントはデータを読み込むべき妥当な入力 XML ファイルを持たないため、この振る舞いは予期されます。問題を解決するには、 コンポーネントのヘッダーをダブルクリックし、編集して、妥当な入力 XML ファイルを提供します。

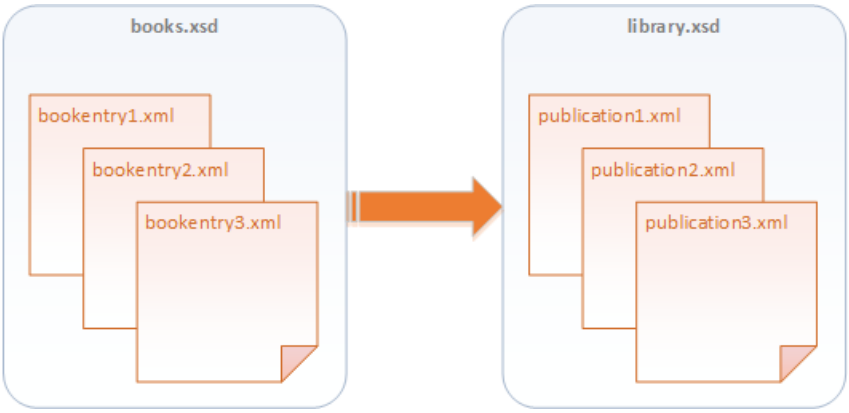


複数のターゲット コンポーネントを持つマッピングのデザインが終わると、このチュートリアルの目的である、各ターゲットの出力を確認して保存することができます。パススルー コンポーネントに関しての詳細は、 チェーンマッピング / パススルーコンポーネントを参照してください。

3.4 ファイルを動的に処理と生成する

このチュートリアルは、複数のソースXML ファイルからデータを読み込み、同じ変換内で複数のターゲットファイルに書き込む方法を説明します。このテクニックを説明するために、次の目的を持つマッピングを作成します：

1. 同じディレクトリ内の複数のXML ファイルからのデータを読み込む。
2. 新規のXML スキーマに各ファイルを変換する。
3. 各ソースXML ファイルのために、新しいスキーマの下で新規ターゲットファイルを生成する。
4. 生成されたファイルからXML と名前空間宣言を削除する。



データ変換の抽象的なモデル

3つのソースXML ファイルをサンプルとして使用します。ファイルの場所は**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーで確認することができ**bookentry1.xml**、**bookentry2.xml**, と **bookentry3.xml** という名前が付けられています。これらのファイルはそれぞれ単一のブックを保管します。

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="1">
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <category>Fiction</category>
      <year>1876</year>
   </book>
</books>
```

bookentry1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
</books>
```

bookentry2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="3">
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <category>Fiction</category>
      <year>1851</year>
   </book>
</books>
```

bookentry3.xml

ソースXML ファイルは次のフォルダーで使用することのできる**books.xsd** スキーマを使用します :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial**。ソースファイルを新規 XML スキーマに変換するには、同じフォルダーで見つけることのできる)**library.xsd** スキーマを使用します。変換の後、マッピングは3つのファイルを新しいスキーマに従い生成します。(下のコードリストを参照)。生成されたファイルが以下のようになるよう マッピングを構成します :**publication1.xml**、**publication2.xml**、と**publication3.xml**。XML 宣言と名前空間が削除されなければならない点に注意してください。

```
<library>
   <publication>
      <id>1</id>
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
</library>
```

publication1.xml

```
<library>
   <publication>
      <id>2</id>
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <genre>Fiction</genre>
      <publish_year>1912</publish_year>
   </publication>
</library>
```

publication2.xml

```
<library>
   <publication>
      <id>3</id>
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <genre>Fiction</genre>
      <publish_year>1851</publish_year>
   </publication>
</library>
```

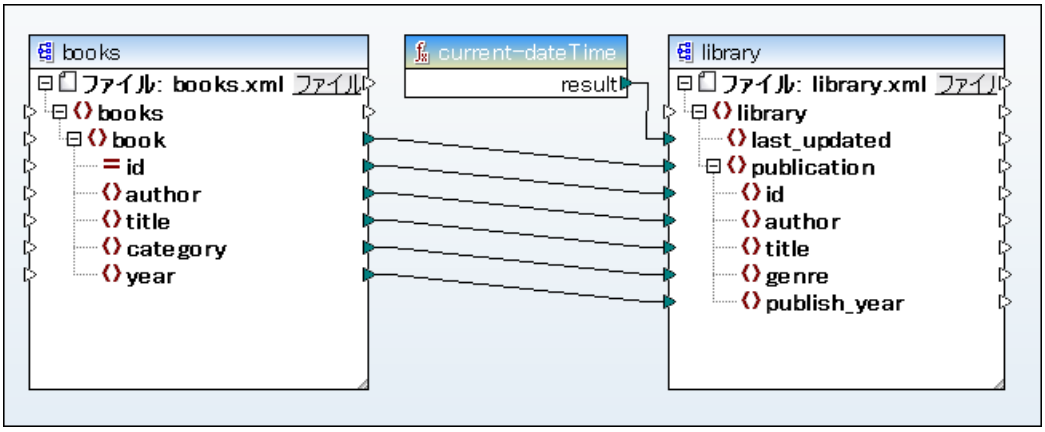
publication3.xml

チュートリアルの目的を達成するために、次のステップを踏みます。

ステップ 1：マッピングデザインファイルの準備

このチュートリアルは、開始のポイントとして**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーから**BooksToLibrary.mfd** マッピングを使用しています。すでにXML を新しいスキーマに変換するチュートリアル内でこのマッピングをデザインしています。開始するには、MapForce で**BooksToLibrary.mfd** ファイルを開き、新しい名前で同じフォルダーに保存します。

> 複数のファイルを参照するため、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内に新しいマッピングが保存されることを確認してください。



BooksToLibrary.mfd (MapForce Basic Edition)

ステップ 2：入力を構成する

MapForce に複数のXML インスタンスファイルを処理するように命令するには、ソースコンポーネントのヘッダーをダブルクリックします。コンポーネント設定ダイアログボックス内に入力ファイルとして**bookentry*.xml** を入力します。



コンポーネント設定 ダイアログボックス

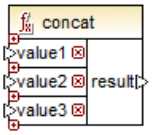
ファイル名内のアスタリスク（*）ワイルドカード文字は、マッピング入力に**bookentry-** プレフィックスを伴う全てのファイルを使用するように命令します。パスは相対的であるため、MapForce はマッピングファイルと同じディレクトリ内の全ての **bookentry-** ファイルを検索します。マッピングファイル.*ワイルドカード文字を使用しているため、必要であれば、絶対パスを入力することもできます。

ステップ 3：出力を構成する

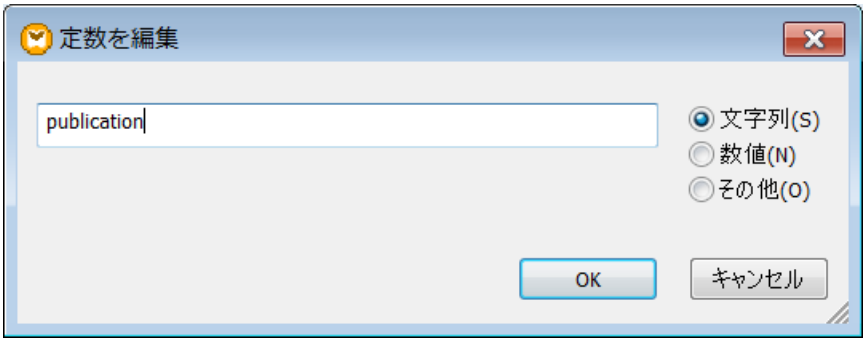
各出力ファイルのファイル名を作成するには、**concat** 関数 を使用します。この関数は引数に与えられた全ての値を連結します。

concat 関数を使用してファイル名を作成する：

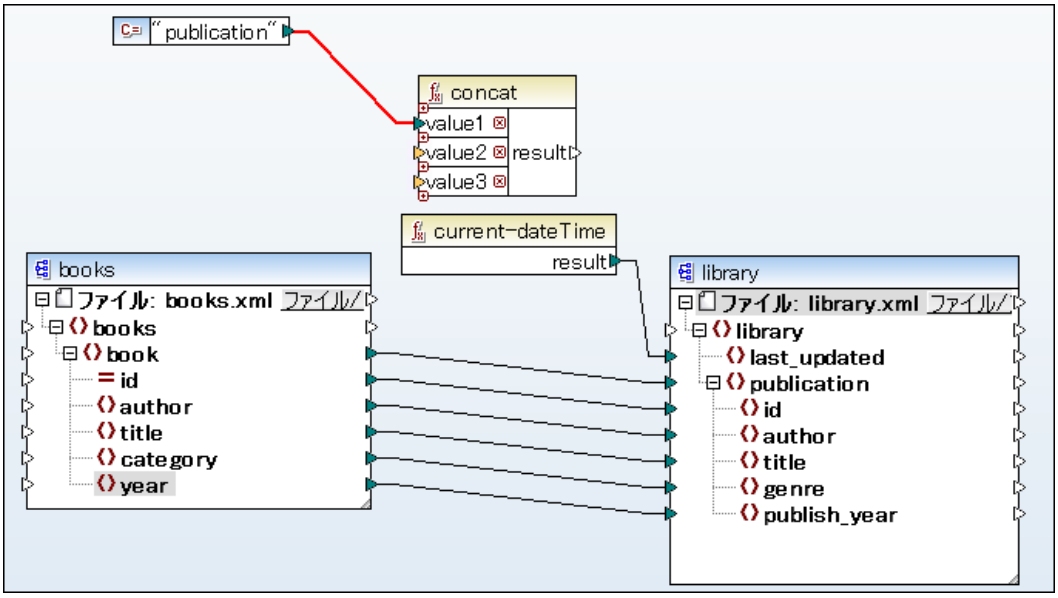
1. **concat** 関数 をライブラリウィンドウで検索し、マッピングエリアにドラッグします。デフォルトでは、この関数はパラメータが2つあるマッピングに追加されます。しかし、必要であれば新しいパラメーターを追加することができます。関数コンポーネント内の **「パラメーターの追加」**（⊡）シンボルをクリックして、3番目のパラメーターを追加します。**「パラメーターの削除」**（⊠）シンボルをクリックするとパラメーターは削除されることに注意してください。



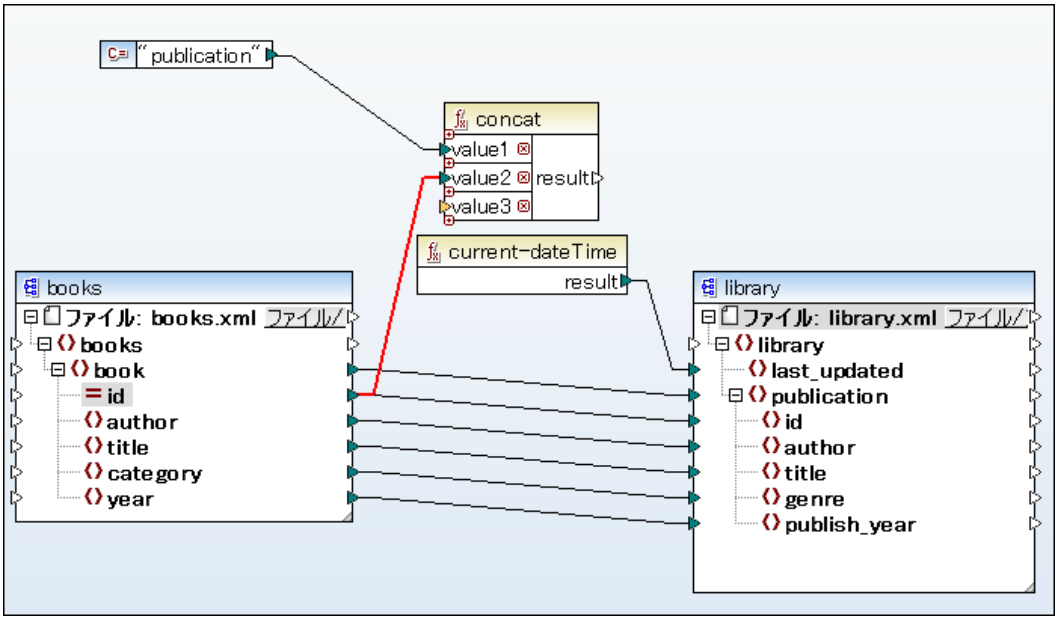
2. 定数を入力するには、（**挿入」** メニューから **定数」** をクリックします）。値を提供するようにプロンプトされると、"publication" を入力して、**文字列** オプションを変更しないでそのままにしておきます。



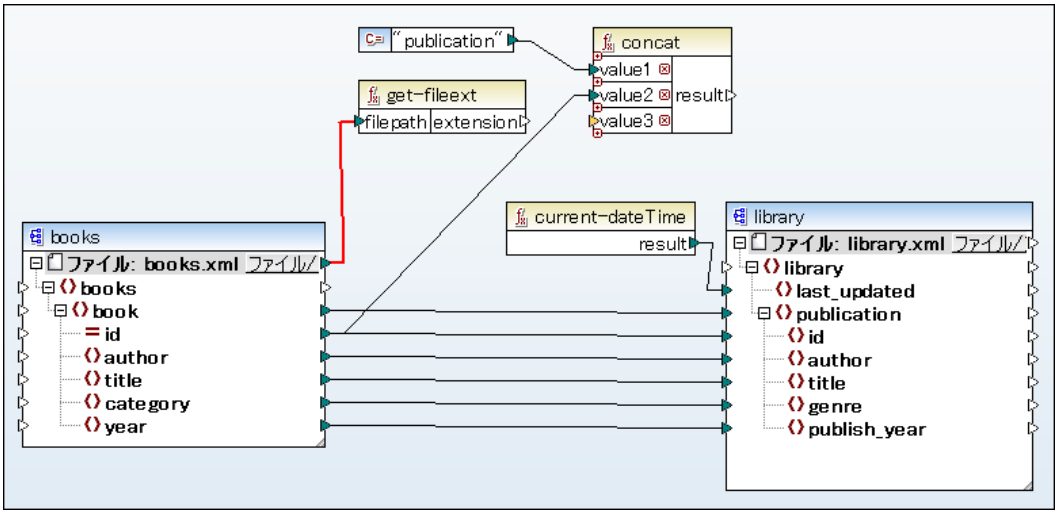
3. **concat** 関数の**value1** と定数をと接続します。



4. ソースコンポーネントの `id` 属性をconcat 関数の**value2** と接続します。



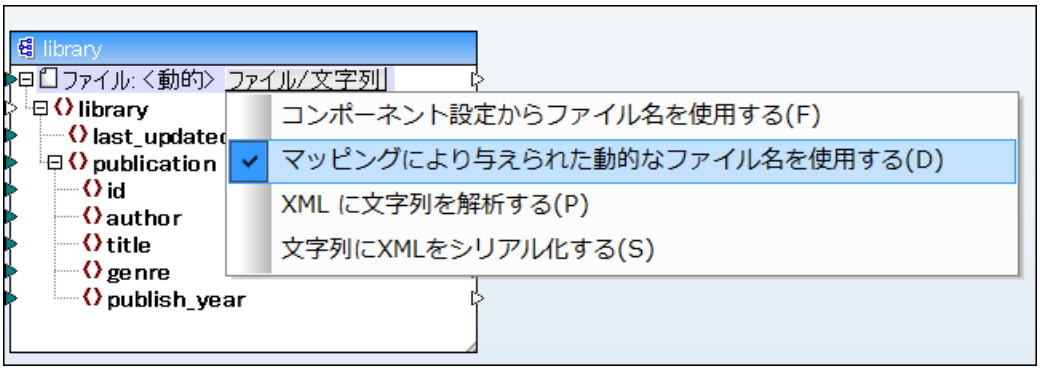
5. ライブラリウィンドウ内で`get-fileext` 関数 を検索し、マッピングエリアにドラッグします。ソースコンポーネント (**File: books.xml**) のトップノードからこの関数の**filepath** パラメータへ接続を作成します。`get-fileext` 関数の結果から`concat` 関数の**value3** への接続を作成します。これを行うことにより、ソースファイル名から拡張子の部分 (この場合、 xml) を抽出することができます。



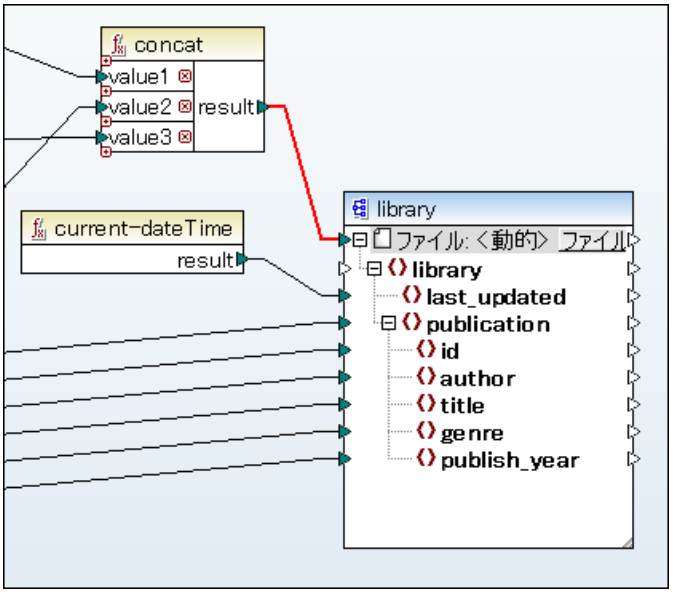
concat 関数のパラメーターに、連結されると生成されるファイル名（例：**publication1.xml**）を作成する3つの値を提供しました：

パート	サンプル
定数 "publication" は定数文字列の値 "publication" を与えます。	publication
ソースXML ファイルの属性 id は、各ファイルのための一意の識別しの値を与えます。これは全てのファイルが同じ名前で生成されないようにするためです。	1
get-fileext 関数 は、生成されるファイルの拡張子を返します。	.xml

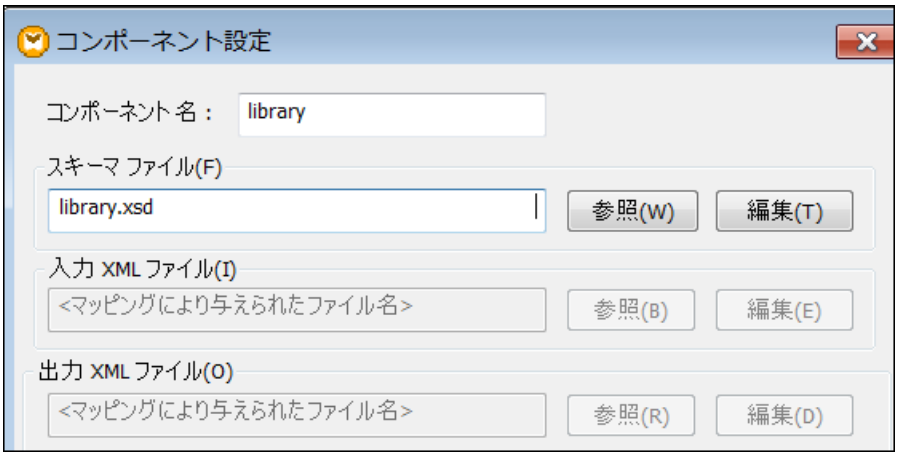
MapForce にマッピングが実行される際に実際にファイル名を作成するように命令することができます。これを行うには、**「ファイル」**（File）をクリックする、または ターゲットコンポーネントの **「ファイル文字列」**（File/String）ボタンから **「マッピングから与えられた動的なファイル名を使用する」** を選択します



マッピングにより与えられた名前を伴うMapForce にインスタンスファイルを動的に生成するように命令しました。この特別な例では、**concat** 関数により名前が作成されました。ですから、**concat** 関数の結果とターゲットコンポーネントの **File: <dynamic>** ノードを接続します。

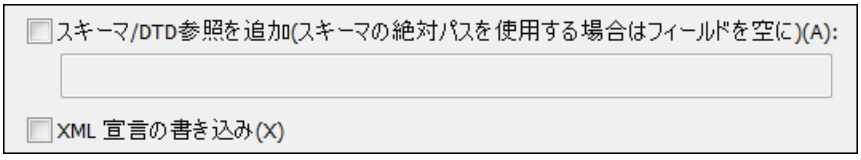


この時点でターゲットコンポーネントヘッダーをダブルクリックすると、**「入力 XML ファイル」**と**「出力 XML ファイル」**テキストボックスが無効化されており、値が**<マッピングにより与えられたファイル名>**を示していることに気がつくはずです。



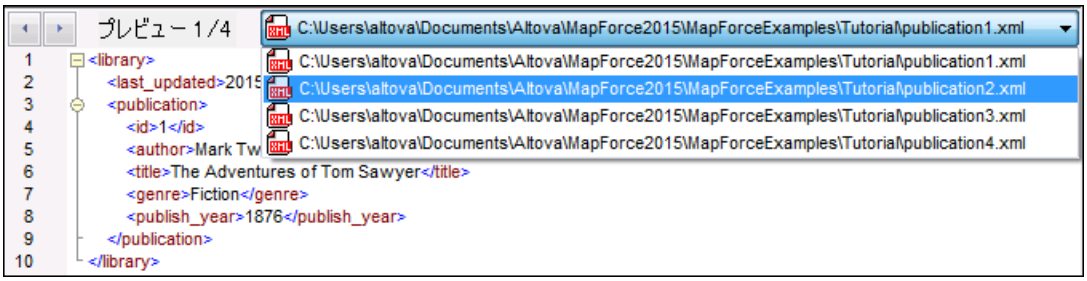
これは、インスタンスファイル名を動的にマッピングから提供したことを示し、コンポーネント設定内で定義することは関連性がありません。

最後に、ターゲットからXML 名前空間とスキーマ宣言を削除する必要があります。これを達成するには、コンポーネント設定ダイアログボックスの**「スキーマ /DTD レファレンスの追加 ..」**と**「XML 宣言の書き込み」**チェックボックスから選択を解除します。



結果と生成されたファイル名を確認するために、マッピングを実行します。 このマッピングは複数の出力ファイルを生成します。出力ペインの左上の左と右ボタンを使用して出力ファイル全体を移動することができます。 または横にあるドロップダウ

ンリストからファイルを選択します。



Chapter 4

一般的なタスク

4 一般的なタスク

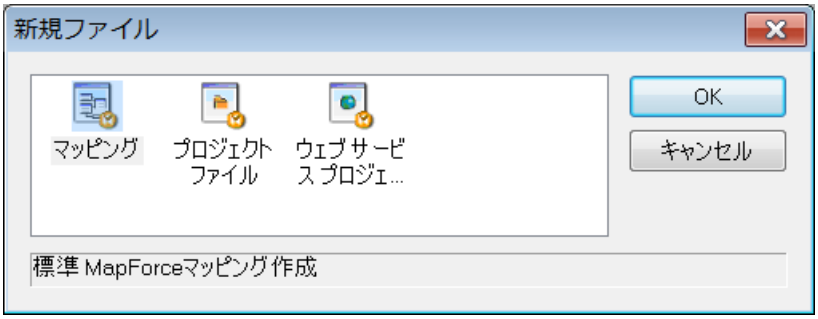
このセクションは、マッピングとの作業、コンポーネント、接続、マッピングプロジェクト、などのMapForce の全般的なタスクとコンセプトを説明します。

4.1 マッピングとの作業

MapForce マッピングデザイン (または、略して "マッピング") は、データがどのように1つのフォーマットから他のフォーマットに変換されるかの視覚的な表現です。マッピングは、データ変換を行うためのMapForce マッピングエリア内で1つのスキーマから他のスキーマに追加するコンポーネントから構成されています。 (例 :XML ドキュメントを1つのスキーマから他のスキーマに変換)。有効なマッピングは、1つまたは複数のターゲットコンポーネントに接続されている、1つまたは複数の複数のソースコンポーネントから構成されています。マッピングを実行して、結果を直接 MapForce 内で確認することができます。コードを生成し、外部で実行することも可能です。MapForce 実行可能ファイルにマッピングをコンパイルし、MapForce Server またはFlowForce Server を使用してマッピングの実行を自動化することもできます。 MapForce は、マッピングを .mfd の拡張子を持つファイルとして保存します。

新規マッピングの作成方法 :

1. 以下を行います :
   - o **「ファイル」** メニューから **新規作成」** をクリックします。
   - o **新規作成」** ( ) ツールバーボタンをクリックします。
2. **「マッピング」** をクリックして、 **「OK」** をクリックします。



マッピングが作成されました。しかしながら、空のため何も起こりません。最低でも妥当なマッピングは2つの接続されたコンポーネントを必要とします。ですから、次のステップではマッピングにコンポーネントを追加し (マッピングにコンポーネントを追加するを参照) 2つのコンポーネントの間に接続を描きます。(接続との作業を参照)。

4.1.1 マッピングにコンポーネントを追加する

MapForce 内では、 "コンポーネント" という用語はデータの構造 (スキーマ) またはデータがどのように変換 (関数) されるかを視覚的に表します。コンポーネントはすべてのマッピングを構築する中心的な役割を果たします。以下は、MapForce コンポーネントの例です :

- 定数
- データベース
- フィルター
- 条件
- フラットファイル (CSV、固定長と他のテキストファイル)
- 関数コンポーネント
- EDI ドキュメント (UN/EDIFACT、ANSI X12、HL7)
- Excel 2007+ ファイル
- 単純型 入力コンポーネント
- 単純型 出力コンポーネント
- 変数

- XBRL ドキュメント
- スキーマおよびDTD

マッピングにコンポーネントを追加するには以下を行います：

- **「挿入」**メニューから追加するコンポーネントの型 (例：**「XML スキーマ/ファイル」**) をクリックします。
- Windows ファイルエクスプローラーからファイルをドラッグしてマッピングエリアにドロップします。このオペレーションは互換性のあるファイルベースのコンポーネントのみに対して使用することができます。
- コンポーネントの挿入ツールバー内の適切なボタンをクリックします。



コンポーネントの挿入ツールバー (MapForce Enterprise Edition)

それぞれのコンポーネントの型には、特定の目的と振る舞いがあります。コンポーネントの型が必要な箇所では、MapForce は、コンテキストウィザードステップまたはダイアログボックスで手順を踏んで、説明を行います。例：XML スキーマを追加する場合、通知ダイアログボックスがインスタンスファイルをオプションで選択するように促します。

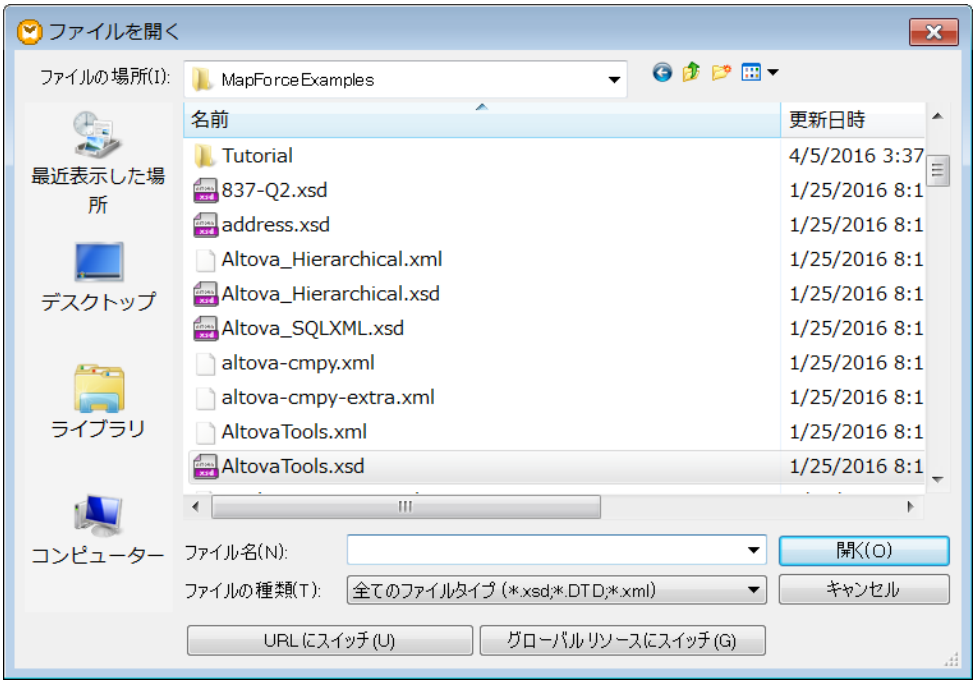
コンポーネントについての説明は、 コンポーネントとの作業を参照してください。マッピングソースまたはターゲットとしてサポートされる個々の技術に関しては、データソースとターゲットを参照してください。データを一時的に保管または変換するMapForce ビルドインコンポーネントに関しては、または (フィルタリングまたは並べ替えなどの) マッピングのデザインを参照してください。

4.1.2 URL からコンポーネントを追加する

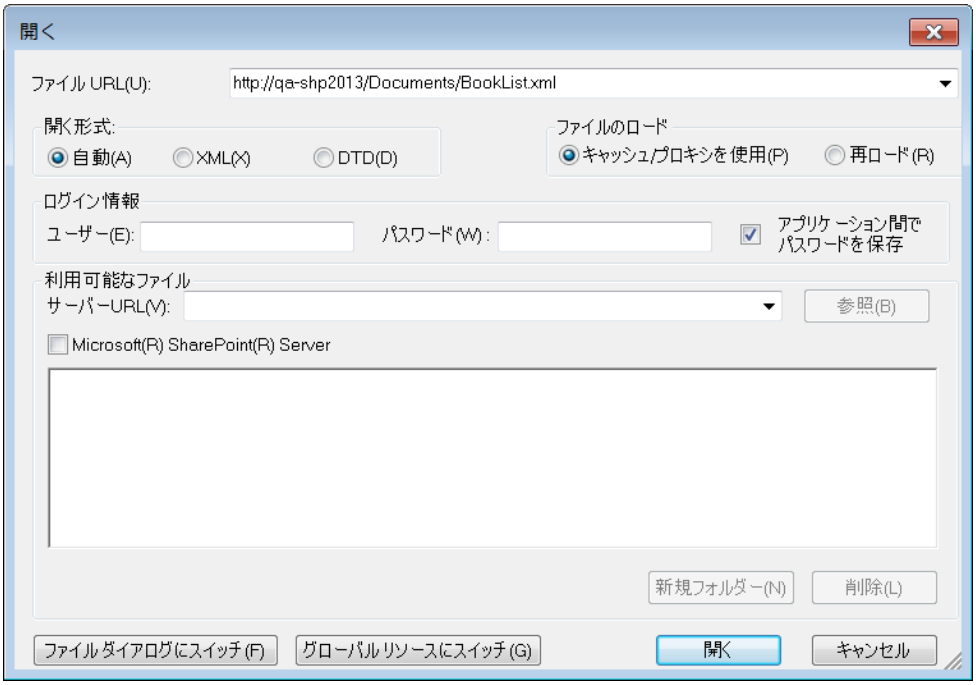
マッピングコンポーネントとしてローカルファイルを追加するだけでなく、URL からファイルを追加することができます。この操作はコンポーネントをソースコンポーネントとして追加した場合のみサポートされます。 (すなわち、マッピングは、リモートファイルからデータを読み込みます)。

URL からコンポーネントを追加する：

1. **「挿入」**メニューから追加するコンポーネントの型を選択します (例：**「XML スキーマ/ファイル」**)。
2. **「開く」**ダイアログボックスから **「URL にスイッチ」** をクリックします。



3. **「ファイル URL」**テキストボックスにファイルの URL を入力して、**開く**をクリックします。



「ファイル URL」テキストボックス内のファイル型がステップ 1 で指定されたファイル型と同じであることを確認してください。

サーバーがパスワード認証を必要とする場合、ユーザー名とパスワードを入力するようにプロンプトされます。次回 MapForce を起動する際に使用できるように、ユーザー名とパスワードを保存するには、「開く」ダイアログボックス内の「ア

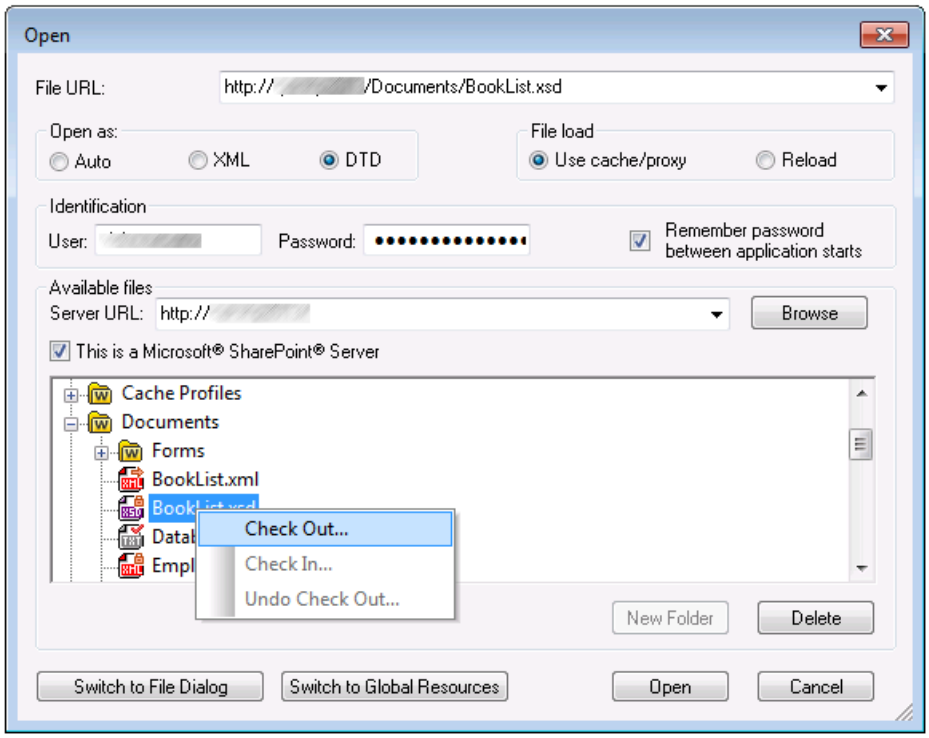
「アプリケーション間でパスワードを保存」チェックボックスを選択します。

「開く形式」設定は、ファイルを開く際のパーサーのための文法を定義します。デフォルトと奨励されるオプションは**「自動」**です。

アップロードするファイルに変更が加えられない場合、**「キャッシュ/プロキシの使用」**オプションを選択して、データをキャッシュし、ロードをスピードアップします。それ以外の場合、マッピングを開くたびにファイルを再ロードする場合は、**「再ロード」**を選択します。

Web Distributed Authoring and Versioning (WebDAV) をサポートするサーバーに関しては、**「サーバー URL」**テキストボックスにサーバーURL を入力した後にファイルを**「参照」**をクリックして、参照することができます。参照は全てのファイルの型を表示しますが、上記のステップ 1 で指定されたファイル型と同じ型が選択されていることを確認してください。それ以外の場合、エラーが発生します。

サーバーがMicrosoft SharePoint Server の場合、**「これは Microsoft SharePoint Server です」**チェックボックスをチェックしてください。これを行うことにより、プレビューエリア内のファイルのチェックインとチェックアウトの状態が表示されます。MapForce を使用中に他のユーザーがファイルを編集できないようにするためには、ファイルを右クリックして、**「チェックアウト」**を選択します。ユーザーにより以前チェックアウトされたファイルをチェックするには、ファイルを右クリックして、**「チェックイン」**を選択します。



開くダイアログボックス（URL に切り替える モード）

4.1.3 データストリーミングについて

データストリーミングとは、任意に大きなデータソースをマッピングの入力または出力として使用を許可するMapForce 内蔵のメカニズムです。データストリーミングはMapForce により生成されたコード内のストリームオブジェクトと間違わないように注意してください。後者はMapForce により生成されたコードをカスタムのC# とJava アプリケーションと統合するためのデータの扱いを指します）。

データストリーミングは次のデータソースに適用することができます：

- XML ファイル
- CSV ファイル
- 固定長 ファイル
- データベース

マッピング内で上記のデータソースを入力または出力として持つ場合、MapForce は、メモリに全てのデータをロードするかわりに、データソースをデータのオープンストリームとして扱い、そのコンテンツを個別に処理します。

メモ： BUILT-IN を変換言語として選択している場合のみデータストリーミングの使用は可能です (変換言語の選択 を参照)。

メモリ使用の注意点

データストリーミング候補であるマッピング入力と出力と作業する場合、マッピングがランダムな入力ソースへのアクセスを必要とする場合、"メモリ不足 "が発生する可能性があります。

例：マッピングがソースデータに`group-by` 関数を適用するコンポーネントを含むと仮定します。`group-by` 関数を入力ファイルのツリー構造全体に適用した場合、メモリにロードされるソールファイル全体が必要となり、この結果、ファイルストリーミングは使用できなくなります。並べ替えなどのマッピングソース全体がメモリにロードされることを必要とする操作には同様が適用されます。

上記のようなシチュエーションが発生した場合、十分な仮想メモリとディスクスペースがシステム上に存在すると変換は完了します。

4.1.4 変換言語の選択

データマッピングの必要条件を満たすために、MapForce には、異なる変換言語を選択する機能が搭載されています。

デフォルトでは、MapForce は、他の言語でサポートされると同様の変換を行う強力なビルドインエンジンを搭載しています。MapForce マッピングを MapForce Server にデプロイする際、ビルドインエンジンは外部のプロセッサーを必要とすることなく実行します。更に、最小または手動介入をデータ変換プロセスで必要とする場合、FlowForce Server を使用して、スケジュールされたジョブを使用してマッピング処理を自動化することができます。

使用するデータにとって最も適切な方法を決定した後に変換言語を選択することが奨励されます。使用することのできる変換言語 は、以下のとおりです：

- BUILT-IN (MapForce により使用されるデフォルトのネイティブ変換エンジン)
- C++
- C#
- Java
- XQuery
- XSLT 1.0
- XSLT 2.0

変換言語を選択するには、以下を行います：

- **「出力」** メニューから変換に使用する言語をクリックします。
- 言語選択ツールバー内の言語名をクリックします。



メモ: マッピング入力と出力の一部は特定の言語ではサポートされていません。例：マッピング入力または出力として、データベースを使用する場合、XSLT コードを生成することができません。ですから、コードを生成する場合、または選択された言語によりサポートされないソースまたはターゲットを持つマッピングの出力をプレビューする場合、MapForce は適切な通知メッセージを表示します。

ビルドイン使用のオプション

BUILT-IN ( ) をマッピングの変換言語として選択した場合、MapForce は、内部の変換エンジンを使用してデータマッピングを実行します。選択された変換言語が Java、C#、または C++ の箇所のマッピングの出力をプレビューする場合、MapForce はこのオプションを使用します。

以下の場合は、変換言語を BUILT-IN に設定することが奨励されています：

- デフォルトのオプションとして、データを変換するために特定の言語を使用する必要の無い場合。
- 大きなファイルを処理し、メモリの使用が注意点な場合。
- 「出力 XML ファイル」内でデジタル署名を生成する必要がある場合。

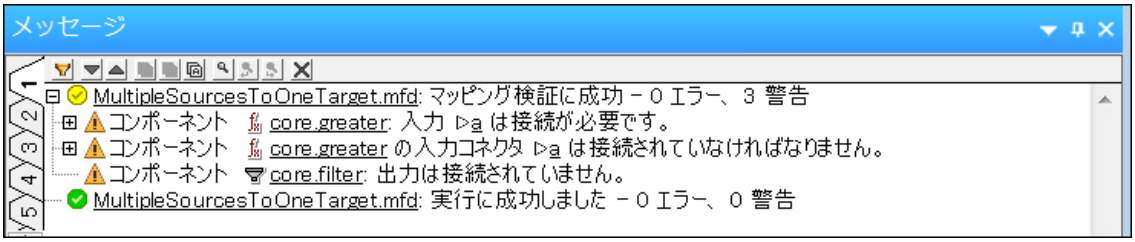
4.1.5 マッピングの検証

MapForce は変換結果をプレビューするために **「出力」** タブをクリックすると、マッピングを自動的に検証します。結果をプレビューする前にマッピングのみを検証することもできます。この機能は、マッピングのエラーと警告をマッピングを実行する前に認識し修正するために役にたちます。マッピングの実行は、処理されたデータにより更なるランタイムエラーまたは警告を生成する場合があります。 例 属性にマップされた値の上書きなど。

マッピングを正確に検証するには、以下を行います：

- **「ファイル」** メニューから **「マッピングの検証」** をクリックします。
- **「検証」** ( ) ツールバーボタンをクリックします。

「メッセージ」ウィンドウは、検証の結果を表示します。例：



メッセージウィンドウ

マッピングを検証するには、MapForce は、マッピングの妥当性をチェックし、（無効または見つからない接続、サポートされないコンポーネントの種類などの）検証結果は、以下のステータスアイコンと共にメッセージウィンドウに表示されます：

アイコン	意味
	変換は成功しました。

アイコン	意味
	変換は警告と伴い完了しました。
	変換は失敗しました。

「メッセージ」ウィンドウは以下のメッセージの型を表示する場合があります：情報メッセージ、警告、とエラー。

アイコン	意味
	情報メッセージを表示します。情報メッセージはマッピングの実行を停止しません。
	警告メッセージを表示します。警告はマッピングの実行を停止しません。警告メッセージは以下の場合に生成される可能性があります。例：必須の入力コネクタへの接続が作成されなかった場合。このような場合、出力は、有効な接続が存在する場所で、これらのコンポーネントのために生成されます。
	エラーを表示します。エラーが発生すると、マッピングの実行に失敗すると、出力は生成されません。XSLT または XQuery コードのプレビューを行うこともできません。

マッピングエリア内の情報、警告または、エラーメッセージをトリガーしたコンポーネントまたは構造をハイライトするには、メッセージウィンドウ内の下線のついたテキストをクリックします。

データを変換するコンポーネントに関しては、（関数または変数など）MapForce 検証は以下のように動作します：

- 必須の**入力コネクタ**が接続されていない場合、エラーメッセージが生成され、変換は停止されます。
- **出力コネクタ**が接続されていない場合、警告が生成され、変換プロセスは続行されます。問題のあるコンポーネントとそのデータは無視され、ターゲットドキュメントにマップされません。

個別のタブ内にそれぞれの検証の結果を表示するには、メッセージウィンドウの左横にある番号の付いたタブをクリックします。これはとても役にたちます。例：複数のマッピングファイルと同時に作業する場合。

メッセージウィンドウ内の他のボタンにより以下のアクションを実行することができます：

- メッセージを型別にフィルターします（例：エラーまたは警告のみを表示）。
- エントリを上下に移動します。
- メッセージテキストをクリップボードにコピーします
- ウィンドウ内で特定のテキストを検索します
- メッセージウィンドウをクリアします

メッセージウィンドウの全般的な情報に関しては、ユーザーインターフェイスの概要を参照してください。

4.1.6 マッピング出力の検証

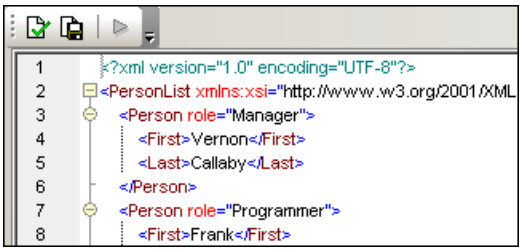
「出力」タブをクリックした後、マッピングをプレビューする場合は、「出力」ペインで出力の結果を確認することができます。この出力を関連するスキーマに対して検証することができます。例：マッピング変換が XML ファイルを生成する場合、結果 XML ドキュメントは XML スキーマに対して検証されます。ターゲットコンポーネントが EDI ファイルの場合、出力は EDI 仕様に対して検証されます（EDI コンポーネントの検証を参照）。同様に、ターゲットが JSON オブジェクトの場合、出力は JSON スキーマに対して検証されます。

XML ファイルに関しては、コンポーネント設定ダイアログボックスの**「スキーマ/DTD レファレンスの追加」**フィールド内のインスタンスファイルに関連するスキーマを指定することができます（XML コンポーネント設定を参照）。生成された XML 出力により参照されるスキーマファイルをパスは指定します。これにより、マッピングの実行時、出力インスタンスが検証されること

ができます。このフィールドにhttp://、絶対または相対パスを入力することができます。 **「スキーマ/DTD レファレンスの追加」**フィールドを選択しない場合、スキーマに対する出力ファイルの検証は不可能です。このチェックボックスを選択し、空白のままにしておくと、検証が終わると、コンポーネント設定ダイアログボックスのスキーマファイル名が出力に生成されます。

コンポーネント設定を開くには、以下を行います：

- **出力の検証」** ツールバーボタンをクリックします。



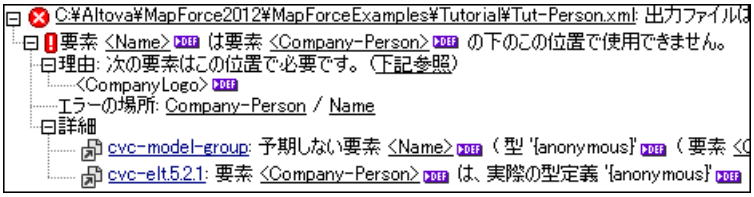
- **出力」** メニューから **出力ファイルの検証」**をクリックします。

メモ: **出力の検証」**ボタンと対応するメニューコマンド（**出力 | 出力ファイルの検証」**）は、出力ファイルがスキーマに対する検証をサポートする場合のみ有効化されます。

検証の結果はメッセージウィンドウ内に表示されます。例：

..\Tutorial\ExpReport-Target.xml: 出力ファイルの検証が成功しました。- 0 エラー、

検証に失敗した場合、メッセージは発生したエラーの詳細情報を表示します。



検証メッセージは更に詳しい詳細情報を含む多くのハイパーリンクを含みます。：

- ファイルパスをクリックすると、MapForce の **出力」** タブの結果の出力が開かれます。
- <ElementName> リンクをクリックすると、 **出力」** タブ内の要素がハイライトされます。
- アイコンをクリックすると、XMLSpy（インストールされている場合）内の要素の定義が開かれます。
- 詳細サブセクションのハイパーリンク例：cvc-model-group）をクリックすると、http://www.w3.org/ Web サイト上の対応する検証のルールについての詳細が開かれます。

4.1.7 出力のプレビュー

MapForce マッピングと作業する場合、結果出力を、外部のプロセッサーまたはコンパイラーを使用して生成されたコードを実行またはコンパイルすることなく、プレビューすることができます。全般的には、生成されたコードを外部で処理する前に、MapForce 内で変換出力をプレビューすることはよい考えです。

マッピングの結果をプレビューする場合、MapForce は、マッピングを実行して、結果出力を出力ペインに表示します。

出力ペインにデータが準備されると、必要に応じて検証し保存します（マッピング出力の検証を参照）。また、 **検索」**コマンドを使用して、（**Ctrl + F** キーの組み合わせ）簡単に特定のテキストのパターンを出力ファイル内で検索することができます。

マッピングの実行に関するエラー、警告、情報メッセージはメッセージウィンドウに表示されます（ユーザーインターフェイスの概要を参照）。

変換の出力をプレビューする方法：

- マッピングウィンドウ下の **出力** タブをクリックします。MapForce は、言語ツールバー内で選択された変換言語を使用してマッピングを実行し、結果の出力を使用して「出力」ペインを作成します。

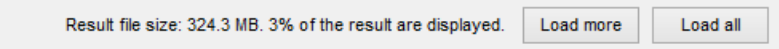
メモ： C++、C#、または Java を変換言語として選択した場合、MapForce は、マッピングをビルドインの変換エンジンで実行します。「出力」ペインに表示される結果は、Java、C++、または C# コードが生成、コンパイル、実行された状態と同じです。

変換の出力を保存するには、以下を行います：

- **出力** メニューから **出力ファイルの保存** をクリックします。
- **生成された出力の保存** ツールバーボタンをクリックします。

部分的な出力のプレビュー

大きな出力ファイルをプレビューする場合、MapForce は、出力ペインに表示されるデータの量を制限します。具体的には、MapForce は、出力ペイン内でファイルの一部を表示します。そして、**更にロード..** ボタンがペインの下の部分に表示されます。**更にロード..** ボタンは現在表示されているデータのファイルの部分の横に表示されます。



メモ： 出力ペイン内にファイル全体がロードされると、**整形出力** ボタンが有効化されます。

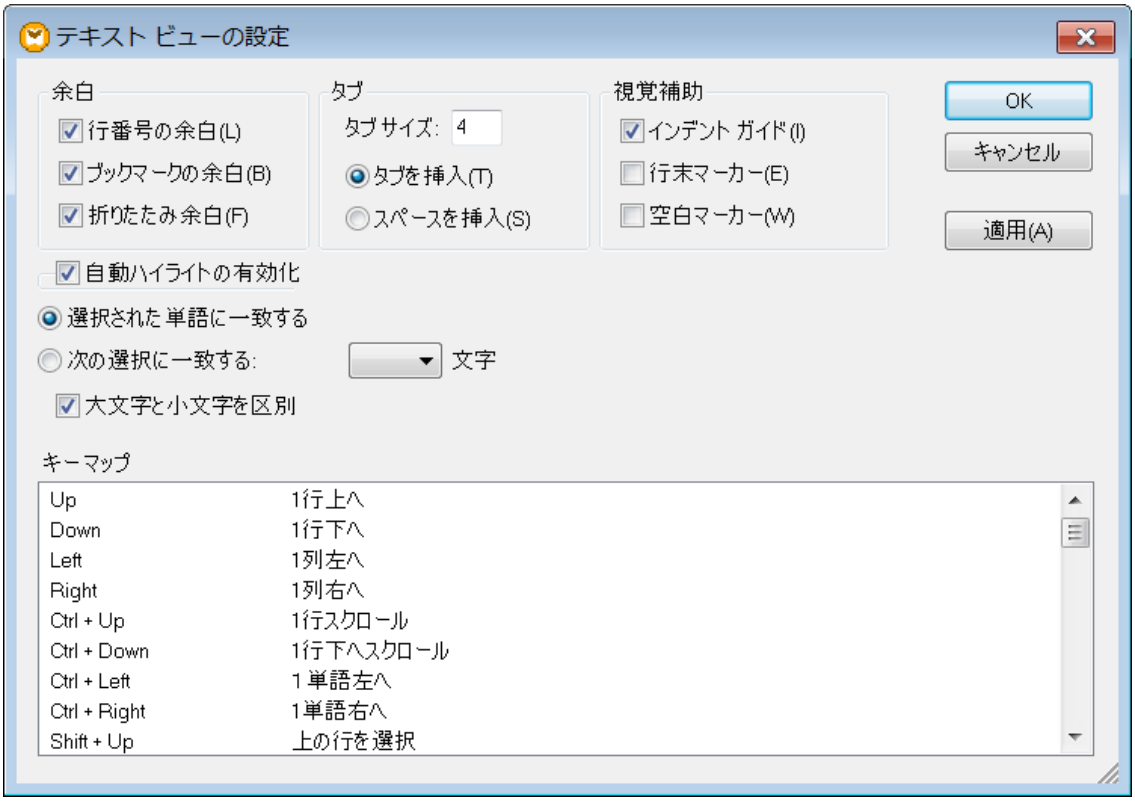
オプション ダイアログボックスの **全般** タブからプレビューの設定を構成することができます（MapForce オプションの変更を参照）。

4.1.8 テキストビューの機能

出力 ペイン、**XSLT** ペイン、および **XQuery** ペインには、テキストをより簡単に表示するための複数の視覚的な補助が搭載されています。以下のような機能が含まれます：

- 行番号
- 構文の色
- ブックマーク
- ソースの折りたたみ
- インデントのガイド
- 行末と空白文字のマーカー
- ズーム
- 整形出力
- ワードラップ
- テキストのハイライト

適用できる箇所では、上記の機能を **テキストビュー設定** ダイアログボックスから切り替え、または、カスタム化することができます。**テキストビュー設定** ダイアログボックス内の設定は、アクティブなドキュメントだけではなく、アプリケーション全体に適用されます。



テキストビュー設定 ダイアログボックス

テキストビュー設定 ダイアログボックスを開くには、以下を行います：

- **出力** メニューから**テキストビュー設定**を選択します。
- **テキストビュー設定** ツールバーボタンをクリックします。
- 出力 ペインを右クリックして、コンテキストメニューから、**テキストビュー設定**を選択します。

ナビゲーションの補助の一部は、テキストビューツールバー、アプリケーションメニュー、キーボードショートカットから切り替えることができます。



テキストビューツールバー

適用することのできるショートカットへの参照は、上記の**テキストビュー設定** ダイアログボックスの「キーマップ」セクションを確認してください。

行番号

行番号は、**テキストビュー設定** ダイアログボックス内で切り替えることのできる行番号 マージンで表示されます。テキストのセクションが折りたたまれると、折りたたまれたテキストの行番号は表示されません。

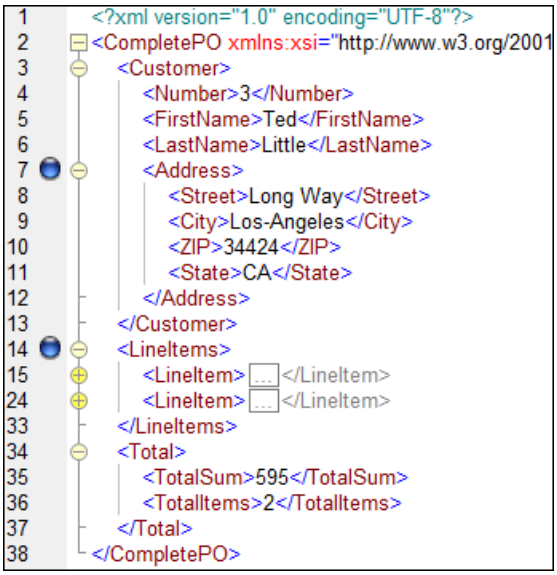
構文の色

構文の色分けはテキストの構文の値に従い適用されます。例えば、XML ドキュメント内では、XML ノードが要素、属性、コンテンツ、CDATA セクション、コメント、処理命令、ノード名（そして一部の場合ではノードのコンテンツ）により異なる

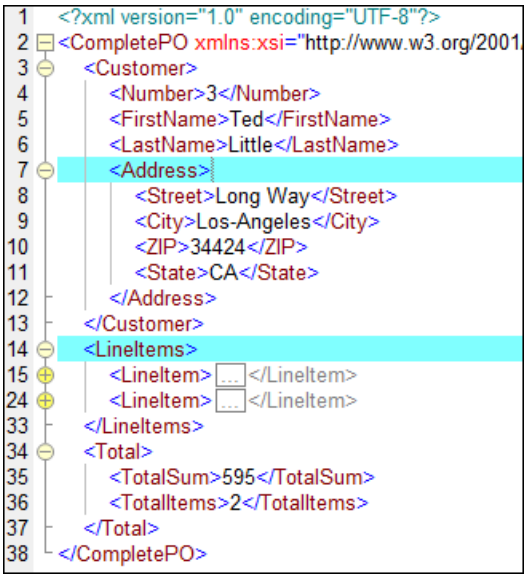
色分けが使用されています。

ブックマーク

ドキュメント内のラインは参照とアクセスのためにブックマークとして使用することができます。ブックマークマージンオンに設定されていると、ブックマークは、ブックマークマージンに表示されます。



それ以外の場合、ブックマークされたラインは水色でハイライトされています。



ブックマークマージンは、**テキストビュー設定** ダイアログボックス内で切り替えることができます。

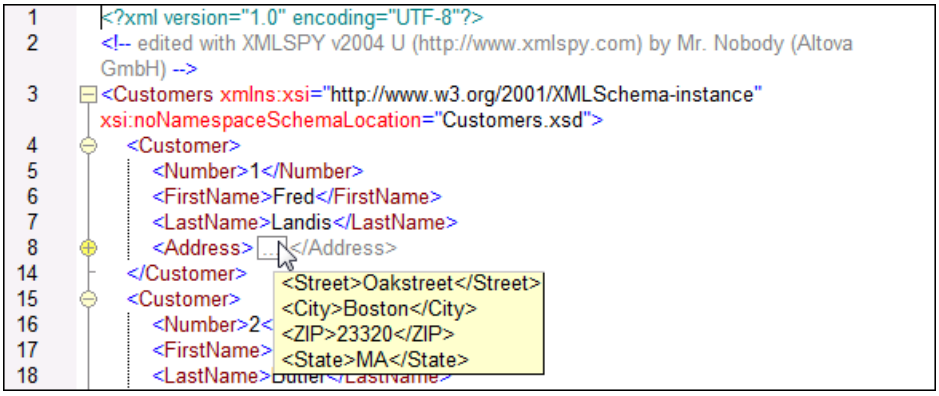
ブックマークを以下のコマンドを使用して編集、および、移動することができます：

- **ブックマークの挿入/削除** (Ctrl + F2)
- **次のブックマークへ移動** (F2)
- **前のブックマークへ移動** (Shift + F2)
- **全てのブックマークを削除** (Ctrl + Shift + F2)

上記の **出力** メニュー内で使用することができます。 **出力** (または、**XSLT**、または、**XQuery**) ペインを右クリックすると、コンテキストメニューを使用してブックマークコマンドを使用することができます。

ソースの折りたたみ

ソースの折りたたみは、ノードの展開と折りたたみ、およびソースの折りたたみマージン内に表示されるノードを表しています。テキストビュー設定 ダイアログボックス内でマージンを切り替えることができます。展開するには、または、テキストの一部を折りたたむには、ウィンドウの左側の "+" と "-" ノードをクリックします。折りたたまれたコードは、省略記号と共に表示されます。これにより、プレビューされているコードを表示するヒントが開かれます。ヒントの終わりには、省略文字が表示されています。



全ての折りたたみの切り替え コマンドは、全てのノードの展開された、または、折りたたまれたフォームを切り替えます。

インデントのガイド

インデントのガイドは、ラインのインデントを示す垂直な点線です。**テキストビュー設定** ダイアログボックス内で切り替えることができます。

メモ: 挿入タブ と **スペースの挿入** オプションは、**出力 | XML テキストの整形出力** オプションを使用すると、効果が与えられます。

行末のマーカーと空白文字のマーカー

行末 (EOL) マーカーと空白文字マーカーは、**テキストビュー設定** ダイアログボックス内で切り替えることができます。下のイメージは、行末と空白文字のマーカーが表示されている状態を表示しています。矢印は、タブ文字を表し、 "CR" は改行を表し、点は空白文字を表しています。

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="books.xsd">
3       <book id="1">
4           <author>Mark Twain</author>
5           <title>The Adventures of Tom Sawyer</title>
6           <category>Fiction</category>
7           <year>1876</year>
8       </book>
9   </books>
```

ズームインとズームアウト

Ctrl キーを長く押して、スクロール (マウスのホイールをスクロール) することによりズームイン、とズームアウトを行うことができます。または、**Ctrl** キーを押しながら "-"、または、 "+ "キーを押してもズームイン、および、ズームアウトすることができます。

整形出力

XML テキストの整形出力 コマンドは、アクティブな XML ドキュメントを構成された形式で表示するために、テキストビュー内で再フォーマットします。デフォルトでは、各子ノードが親ノードより4つの空白文字によりオフセットで表示されます。これは、**テキストビュー設定** ダイアログボックスによりカスタム化することができます。

XML ドキュメントを整形出力するには、**出力 | XML テキストの整形出力** メニューコマンドを選択、または、**整形出力** ツールバーボタンをクリックします。

ワード ラップ

現在アクティブなドキュメント内で、ワード ラップを切り替えるには、**出力 | ワードラップ** メニューコマンドを選択、または **ワードラップ** ツールバーボタンをクリックします。

テキストのハイライト

テキストを選択すると、ドキュメント内で選択されたテキストに一致する箇所が自動的にハイライトされます。選択箇所は薄い青でハイライトされ、一致する箇所は薄いオレンジでハイライトされます。選択箇所と一致する箇所は、灰色のマーカーによりスクロールバー上に表示されます。現在のカーソルの位置は、スクロールバー上で青いマーカーにより表示されています。

テキストのハイライトをオンにするには テキストビュー設定 ダイアログボックス内の **自動ハイライトの有効化** を選択します。選択範囲は、文字全体、または、固定された文字数に設定することができます。大文字と小文字の区別も指定することができます。

文字の選択範囲は、選択範囲の文字の最初の文字から開始する、一致する文字の最小数を指定することができます。例えば、2つ、または、2文字以上の文字が一致するように選択することができます。この場合、1つの文字の選択は、一致とは考えられません。2文字以上の一致のみが一致と考えられます。ですから、この場合、`t` を選択すると、一致は表示されません。`ty` を選択すると、`ty` に一致するが表示されます。 `typ` を選択すると、`typ` に一致するすべてのが表示されます。

文字の検索に関しては、文字を区切るために以下が考慮されます：角かっこのない要素名、要素タグの角かっこ、属性名、引用符無しの属性の値。

4.1.9 テキストビュー内の検索

出力 ペイン、**XQuery** ペイン、および**XSLT** ペイン内のテキストは、広範囲のオプションの組み合わせと視覚的な補助

を用いて、検索をすることができます。

検索を開始するには、**Ctrl+F** を押します (または、メニューコマンド **編集 | 検索** を選択します)。ダイアログ内に入力された検索用語を、ドキュメント全体で、または、選択された範囲で検索することができます。

- 検索する文字列を入力、または、コンボボックスを使用して最近使用した10件の文字列を選択します。
- 検索する文字列を入力、または、選択すると、全ての一致がハイライトされ、スクロールバー内で一致がベージュ色で表示されます。
- 現在選択されている一致は、他の一致と異なる色で表示されます。一致の位置はスクロールバー上で濃い青のカーソルマーカーで表示されます。
- 検索用語フィールド内に一致の総数が、現在選択されている一致のインデックス位置と共に表示されます。例えば、**2 の 4** は、4つ中の2番目の一致が選択されていることを示しています。
- 右下の **前へ** (**Shift+F3**) と **次へ** (**F3**) ボタンを選択することにより、1つの一致から次の一致へ両方向に移動することができます。



検索ダイアログを閉じるには、右上の **閉じる** ボタンをクリック または、**Esc** を押します。

以下の点に注意してください:

- 検索ダイアログにモードは存在しません。これは、検索ダイアログは、テキストビューを使用する場合でも、開き続けることができます。
- ダイアログボックスを開く前にテキストが選択されている場合、選択されたテキストは、自動的に検索用語フィールドに挿入されます。
- 選択範囲内で検索を行うには、以下を行います:(i) 選択範囲をマークします (i) *選択範囲内で検索* オプションをオンにして、検索範囲をロックします。 (ii) 検索用語を入力します。他の選択範囲内を検索するには、現在の選択範囲を、*選択範囲内を検索* オプションをオフにしてアンロックします。そして、新規の選択範囲を*選択範囲内を検索* オプションをオンにして切り替えます。
- 検索ダイアログが閉じられると、現在の検索は **F3** を押すことにより順方向検索し、**Shift+F3** を押すことにより逆方向検索します。検索ダイアログは再度表示されます。

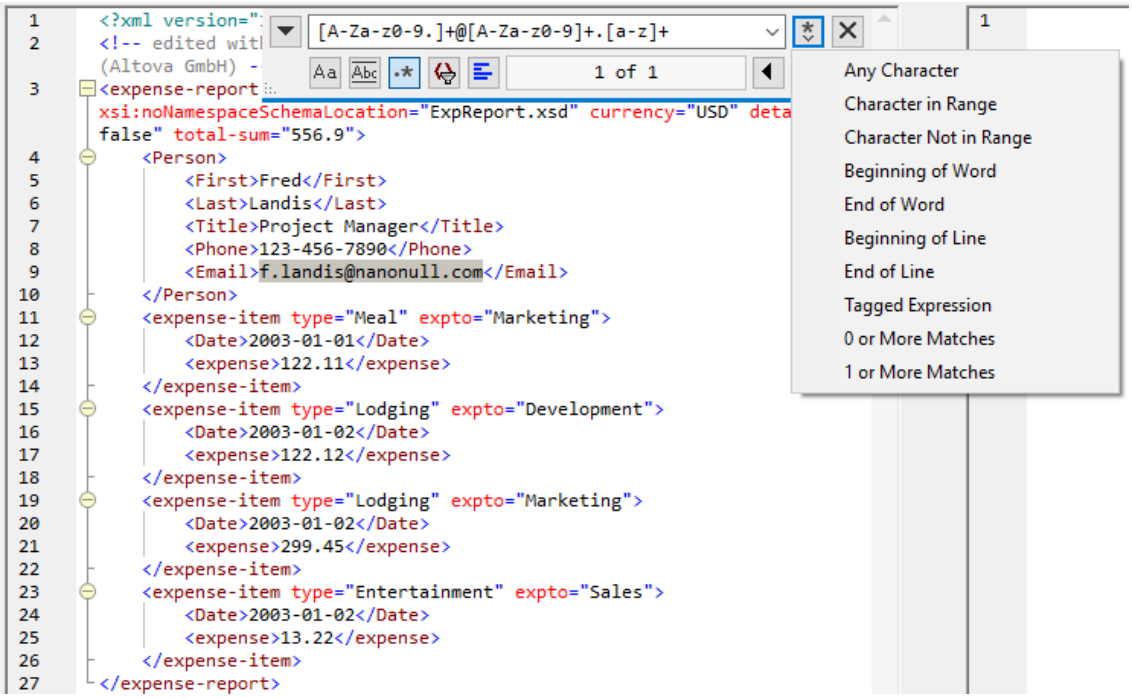
検索オプション

検索の条件は、検索フィールドの下にあるボタンにより指定することができます。オプションがオンになっている場合、ボタンの色は青に変更されます。以下のオプションから選択することができます：

オプション	アイコン	詳細
大文字と小文字を区別する	Aa	切り替えられると、大文字と小文字を区別する検索が行われます（「Address」は「address」とは異なります）。
単語単位で検索	Abc	テキスト内の文字のみが一致されます。例えば、入力文字列 *fit* に対して、**単語単位を一致する**がオンにされていると、単語 *fit* のみが検索文字列に一致します。*fitness* 内の *fit* は、一致しません。
正規表現	.*	オンに切り替えられると、検索用語は、正規表現として読み取られます。「正規表現の使用」を参照してください。
アンカーの検索	I	検索用語が入力されると、ドキュメント内の一致がハイライトされ、一致の内の1つが現在の選択としてマークされます。**アンカーの検索** の切り替えは、最初の選択がカーソルの位置に対して相対的かを決定します。**アンカーの検索** がオンに切り替えられると、現在選択されている一致が選択され、現在のカーソルの場所の次の一致に一致します。**アンカーの検索** がオフに切り替えられていると、現在選択されている一致はドキュメントの最初から数えて最初の一致が一致します。
選択範囲内の検索		オンに切り替えられると、現在のテキストの選択範囲をロックし検索を選択されている範囲に制限します。それ以外の場合、ドキュメント全体が検索されます。テキストの新しい範囲を選択する前に、選択範囲内の検索 オプションをオフに切り替えて現在の選択範囲のロックを解除します。

正規表現の使用

正規表現（regex）を使用して、テキスト文字列を検索することができます。これを行うには、最初に *正規表現* .* オプションをオンに切り替えます。これにより、検索用語フィールド内のテキストが正規表現として評価されるために検索用語内のテキストを指定することができます。次に、検索用語フィールド内に正規表現を入力します。正規表現の作成をヘルプするために、検索用語フィールドの右にある**正規表現 ビルダー** ボタンをクリックします。ビルダー内のアイテムをクリックし、対応する正規表現メタデータを検索フィールドに入力します。下のスクリーンショットは、電子メールアドレスを検索する単な正規表現を示しています。



正規表現メタ文字のカスタムセットは、テキストを検索し置き換える際にサポートされます。

.	全ての文字に一致します。これは単一の文字のプレースホルダです。
\(abc\)	\(and \) メタ文字は、タグ付けされた式の開始と終了をマークします。タグ付けされた式は、後で参照(バックレファレンス)するために一致する箇所をタグ(「記録」)する際に役にたちます。タグ付けされた式は、正規表現内の .NET フレーバーの一致したサブ式(インデックスされたグループ)に類似しています。9個までのサブ式をタグ付け(そして後から参照)することができます 例えば、**\(the\) \1** は、文字列 the the に一致します。この式は、以下のように説明することができます:以前に一致したタグ付けされた箇所が後に続き、スペース文字が後に続く、文字列「the」を一致、(および、タグされた箇所として記録)します。
\n	n が1 から9 の箇所では、n は、最初から9番目のタグ付けされた箇所を指します(上を参照してください)。
\<	単語の先頭に一致します。
\>	単語の末尾に一致します。
\	バックスラッシュが後に置かれている文字をエスケープします。すなわち、式 \x は、文字 x を文字通り使用することができます。例えば、\[は、文字のセットの開始としてではなく、[として解釈されます。
[...]	このセット内の文字に一致します。例えば、**[abc]** は、a、b または c の文字に一致します。範囲を使用することができます:例えば、小文字のために **[a-z]** を使用します。
[^...]	このセット内では内文字に一致します。例えば、**[^A-Za-z]** は、アルファベット文字以外の文字に一致します。
^	(セット内で使用されていない限り)行頭に一致します。
$	行末に一致します。例えば、**A+$** は、行末の A に一致します。

*	前の式のゼロ、または複数の発生に一致します。例えば、**Sa*m** Sm、Sam、Saam、Saaam などに一致します。
+	前の式の1つの、または複数の発生に一致します。例えば、**Sa+m** はSam, Saam, Saaam などに一致します。

特殊文字の検索と
正規表現 オプション .* が有効化されていると想定して、テキスト内の特殊文字を検索することができます

- \t (タブ)
- \r (キャリッジリターン)
- \n (新しいライン)
- \\ (バックスラッシュ)

例えば、タブ文字を検索するには、**Ctrl + F** を押して、 .* オプションを選択し、検索ダイアログボックス内に**\t** を入力します

4.1.10 XSLT コードのプレビュー

XSLT 1.0 またはXSLT 2.0 を変換言語として選択している場合、MapForce により生成されたXSLT コードをプレビューすることができます。 (変換言語の選択を参照)。

変換されたXSLT 1.0 (またはXSLT 2.0) コードをプレビューするには、以下を行います :

- XSLT 1.0 コードをプレビューするには、マッピングウィンドウの下の**XSLT** タブをクリックします。
- XSLT 2.0 コードをプレビューするには、マッピングウィンドウの下の**XSLT2** タブをクリックします。

メモ: XSLT (またはXSLT2) タブは、XSLT (またはXSLT2 をそれぞれ) 変換言語として選択すると使用することができます。

4.1.11 XSLT コードの生成

XSLT コードを生成する:

1. メニュー「ファイル| コードの生成 | XSLT 1.0 (またはXSLT 2.0)」を選択します。
2. 生成されたXSLT ファイルを保存するフォルダーを選択し、 **「OK」**をクリックします。MapForce は、コードを生成し、メッセージウィンドウ内にオペレーションの結果を表示します。

生成されたxslt ファイルの名前には、 **<A>MapTo<B>.xslt** の書式を持ちます :

- "<A>" は、マッピング設定内の**アプリケーション名** フィールドの値です (次を参照してください:マッピング設定の変更).
- "<B>" は、ターゲットマッピングコンポーネントの名前です。この値を変更するには、ターゲットコンポーネントの設定を開き、 **コンポーネント名** フィールドの値を変更してください (次を参照してください:コンポーネント設定の変更)。

XSLT ファイルが保存されるフォルダーは、RaptorXML Server によりデータ変換を行うことができる **DoTransform.bat** というバッチファイルを含みます。

RaptorXML Server を使用して変換を行う:

1. RaptorXML をダウンロードページからダウンロードしインストールします (https://www.altova.com/ja/download-trial-server.html)。
2. 上の操作で指定した出力フォルダーに入りされている**DoTransform.bat** バッチファイルを起動します。

RaptorXML がインストールされた場所を環境変数の Path 変数へ追加する必要がある場合があります。 ウェブサイト上のドキュメントページで RaptorXML ドキュメントを確認することができます (https://www.altova.com/ja/documentation.html)。

4.1.12 XQuery コードのプレビュー

XQuery をデータ変換言語として選択している場合、MapForce により生成された XQuery コードをプレビューすることができます。 (変換言語の選択 を参照)。

生成された XQuery コードをプレビューする方法 :

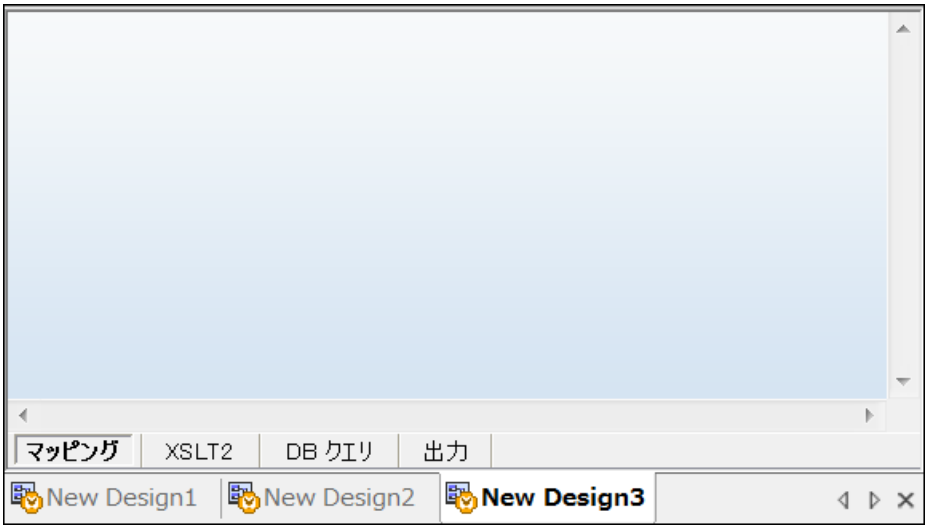
- マッピングウィンドウの下にある **「XQuery」** タブをクリックします。

メモ: XQuery タブは、**XQuery** が変換言語として選択されていると使用することができます。

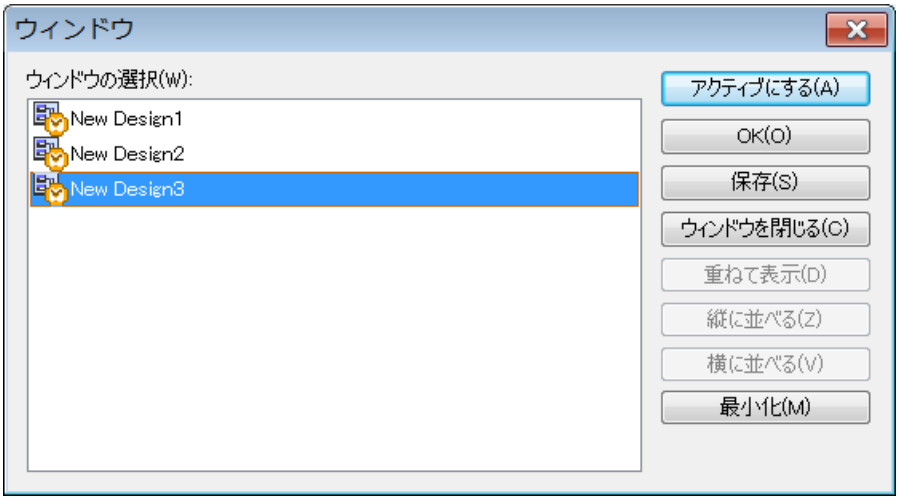
4.1.13 複数のマッピングウィンドウとの作業

MapForce は、複数のドキュメントインターフェイス (Multiple Document Interface) (MDI) を使用します。 MapForce で開く各マッピングファイルには個別のウィンドウがあります。 これにより、複数のマッピングウィンドウと作業、または並べ替えやサイズ調整を MapForce メイン (親) ウィンドウで行うことができます。で行うことができます。全ての開かれているウィンドウを標準ウィンドウのレイアウトを使用して以下のように並べ替えることができます : 上下に並べる、左右に並べる、重ねて表示。

複数のマッピングが MapForce 内で開かれている場合、マッピングペインの下の部分に表示されているタブを使用して素早く切り替えることができます。



ウィンドウ管理オプションは、**「ウィンドウ」** メニューと **「ウィンドウ」** ダイアログボックスで使用することができます。 **「ウィンドウ」** ダイアログボックスから、アクションを実行、または現在開かれているマッピングウィンドウで (保存、閉じる、または最小化を含む) アクションを使用することができます。



ウィンドウダイアログボックス

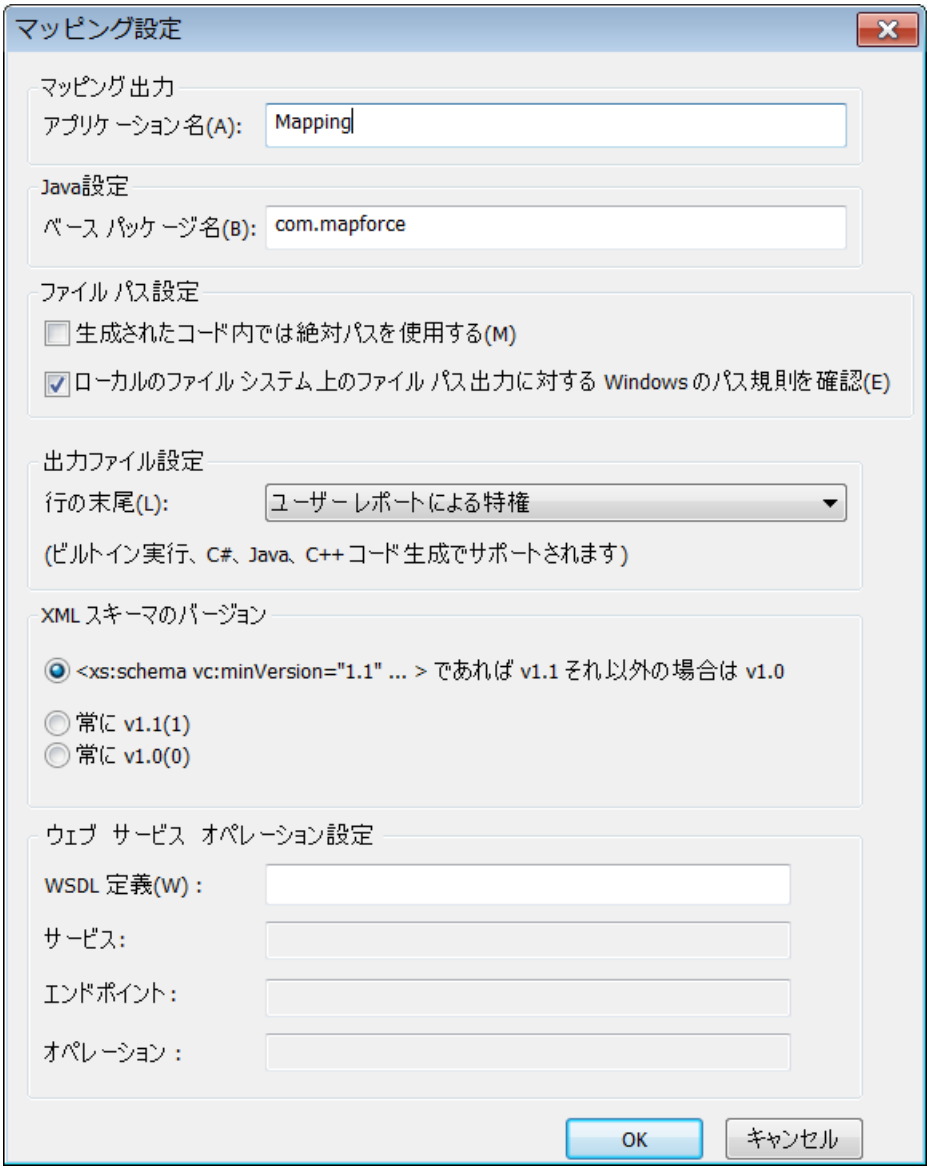
「ウィンドウ| ウィンドウ」を使用して、ウィンドウダイアログボックスを開くことができます。ウィンドウダイアログボックスから複数のウィンドウを選択するには、必要なエントリを **「Ctrl」**キーを押しながらクリックしてください。

4.1.14　マッピング設定の変更

マッピング設定ダイアログボックスから現在アクティブなマッピングデザインファイルのドキュメント特有の設定を変更することができます。この情報は *mfd ファイルに保管されます。

マッピング設定ダイアログボックスの開きかた：

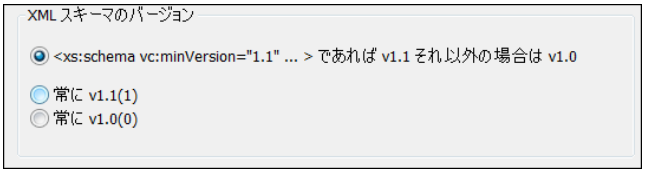
- **「ファイル」**メニューから **「マッピング設定」**をクリックします。



マッピング設定ダイアログボックス

使用することのできる設定は、以下のとおりです。

アプリケーション名	生成される変換ファイルにて使用されるXSLT 1.0/2.0 ファイル名プレフィックス、またはJava、C# またはC++ *アプリケーション名* が定義されます。
ベースパッケージ名	Java 出力のためのベースパッケージ名を定義します。
生成されたコード内でパスを絶対パスにする	生成されたコード内 または、MapForce Server Execution ファイル (mfx) 内で でファイルパスが相対または絶対パスであるかを定義します。詳細に関しては、生成されたコード内のパスについてを参照してください。
Windows パスがファイルパスの規則であることを確認	ローカルのファイルシステム上のファイルパス出力 ...チェックボックスにより、Windows のパス規則が使用されます。XSLT2 またはXQuery) により出力が行われる場合、現在処理されているファイル名は、document-uri 関

	数により取得され、内部的にはローカルファイルに対して file:// というパス形式の URI が使用されます。 このチェックボックスにチェックが入れられている場合、file:// 形式の URI パスが Windows ファイルパス (例 : "C:\...") へ自動的に変換され、それ以降の処理を単純化することができます。
行末	このコンボボックスにより、出力ファイルの行末を指定することができます。「プラットフォーム デフォルト」によりターゲットのオペレーティング システムで使われているデフォルトの設定 (Windows には (CR+LF)、Mac OS X には (LF)、Linux には (LF)) が使用されます。行末は手動でも選択することができます。ここで選択された設定は、異なるオペレーティング システムで動作している FlowForce Server へマッピングをデプロイする際に大きな意味を持つことになります。
XML スキーマ バージョン	マッピング ファイルで使用された XML スキーマ バージョンを定義することができます。バージョン 1.0 または 1.1 に準拠するスキーマを**ロード**するかを定義します。すべてのバージョン 1.1 特有の機能は現在サポートされています。 *xs:schema vc:minVersion="1.1"* 宣言が存在する場合、バージョン 1.1 が使用されます。それ以外の場合は、1.0 が使用されます。  XSD ドキュメントが `vc:minVersion` 属性 または `1.0` または `1.1` 以外の `vc:minVersion` 属性の値を持たない場合、XSD 1.0 がデフォルトのモードです。 **メモ:** `vc:minVersion` 属性 と `xsd:version` 属性を混同しないでください。前者は、XSD バージョン番号を示し、後者はドキュメント バージョン番号を示します。 既存のマッピング内のこの設定を変更することは、選択されたスキーマ バージョンの全ての XML スキーマを再ロードすることを意味し、妥当性を変更する可能性があります。
Web サービス オペレーション設定	マッピング ドキュメントが Web サービス実装の一部である場合、WSDL 定義、サービス、ポート、オペレーション フィールドが自動的に入力されます。

4.2 コンポーネントとの作業

コンポーネントは、MapForce 内のマッピングデザインで中心的な要素です。全般的に、「コンポーネント」という用語は、データソースとデータターゲットとしての役割を果たすオブジェクト、または中間処理の段階にあるマッピング内のデータに簡単な名称を与えるために役に立ちます。

コンポーネントには以下の 2つの主なカテゴリがあります：構造コンポーネントと変換コンポーネント。

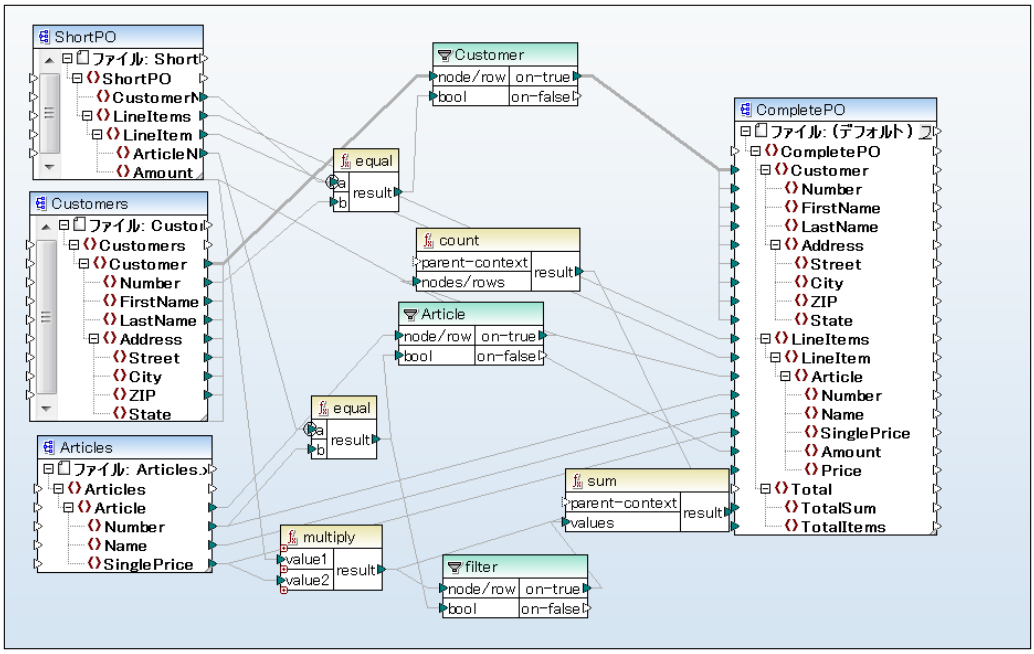
構造コンポーネントは、データの抽象的な構造とスキーマを表します。 例：(メニューコマンド **挿入 | XML スキーマ/ファイル**) を使用して) マッピングエリアにXML ファイルを追加すると、マッピングコンポーネントになります。構造コンポーネントとその特化についての詳細は、データソースとターゲットを参照してください。一部の例外を除いて、構造コンポーネントは、アイテムとシーケンスから構成されます。アイテムはマッピングユニットの最も低いレベルです (例：XML ファイル内の単一の属性、または単純型の要素)。シーケンスはアイテムのコレクションです。

変換コンポーネントは、データを変換 (例：関数)、または変換の手助け (例：定数または変数) をします。これらのコンポーネントを使用して、複数のデータ変換タスクを達成する方法に関しては、マッピングのデザインを参照してください。

構造コンポーネントを使用することにより、ファイルまたはその他のソースからデータを読み込み、また、ファイルまたはその他のソースへデータを書き込み、マッピング処理内の中間の段階で (例：プレビューするために) データを保存することができます。この結果、構造コンポーネントには、以下の3つの型があります：

- ソース。マッピングエリアの左側に配置し、MapForce にデータを読み込むように命令し、コンポーネントをソースとして宣言します。
- ターゲット。マッピングエリアの右側に配置し、MapForce にデータを書き込むように命令し、コンポーネントをソースとして宣言します。
- パススルー。これは特別なコンポーネントの型でソースとターゲットとしての役割を果たします。 (更に詳しい情報に関しては、 チェーンマッピング/パススルーコンポーネントを参照)。

マッピングエリアでは、コンポーネントは三角形で表示されます。次のサンプルマッピングは、 3つのソースコンポーネント、1つのXML コンポーネント、ソースに書き込まれる前にデータが通過する複数の変換コンポーネント (関数とフィルター) を表しています。



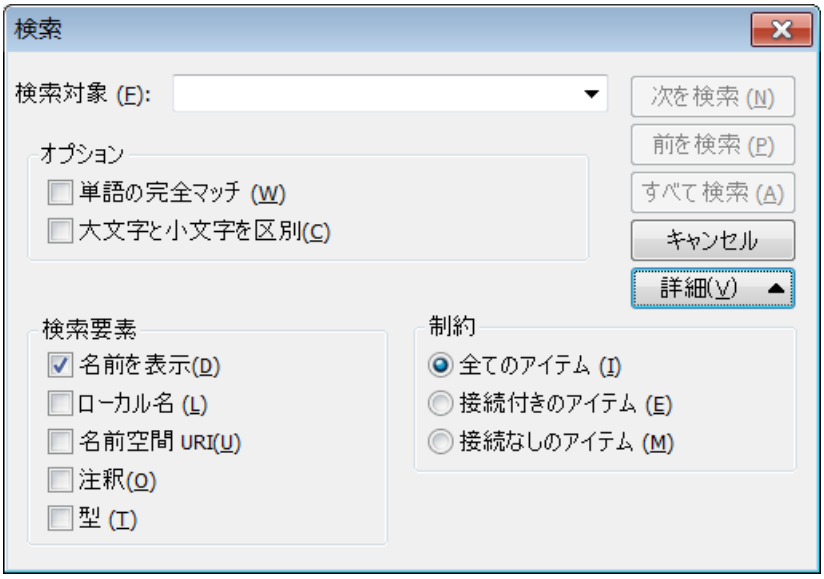
CompletePO.mfd

このマッピングサンプルは以下のパスで見つけることができます：**<Documents>\Altova\MapForce2017\MapForceExamples\CompletePO.mfd**。

4.2.1 コンポーネント内の検索

特定のノード/アイテムをコンポーネント内で検索する方法：

1. 検索するコンポーネントをクリックします。「CTRL+F」キーをクリックします。
2. 検索用語を入力して、**「次を検索」**をクリックします。

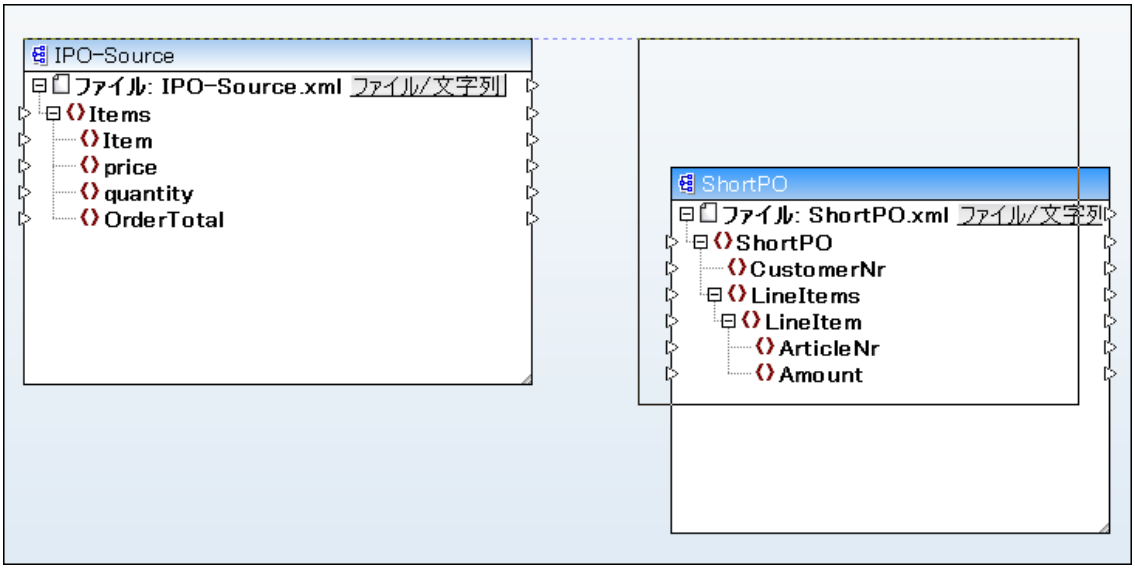


高度なオプションを使用して、どのアイテム (ノード) が検索されるか、また特定の接続をベースにした厳密な検索オプションを定義します。

4.2.2 コンポーネントの整列

コンポーネントをマッピングペイン内でドラッグすると、MapForce は、自動配置 ガイドラインを表示します。このガイドラインは、マッピングウィンドウ内で他のコンポーネントと位置を合わせる際とても役に立ちます。

下のサンプルマッピングでは、下のコンポーネントを移動します。ガイドラインは、マッピングの左のコンポーネントに位置を合わせることができます。



コンポーネント自動配置 ガイドライン

このオプションを有効化または無効化する：

1. **「ツール」**メニューから**「オプション」**をクリックします。
2. **「編集」**グループから、**「マウスのドラッグにてコンポーネントを整列」**のチェックボックスをチェックします。

4.2.3 コンポーネント設定を変更する

コンポーネントをマッピングエリアに追加した後、から適用することのできる設定をコンポーネント設定ダイアログボックスから構成します。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます：

- コンポーネントを選択して、**「コンポーネント」**メニューから**「プロパティ」**をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、**「プロパティ」**をクリックします。

コンポーネントの型により使用できるオプションが異なることにご注意してください。それぞれのコンポーネントの型に適応することのできる設定に関するレファレンスは、以下を参照してください：

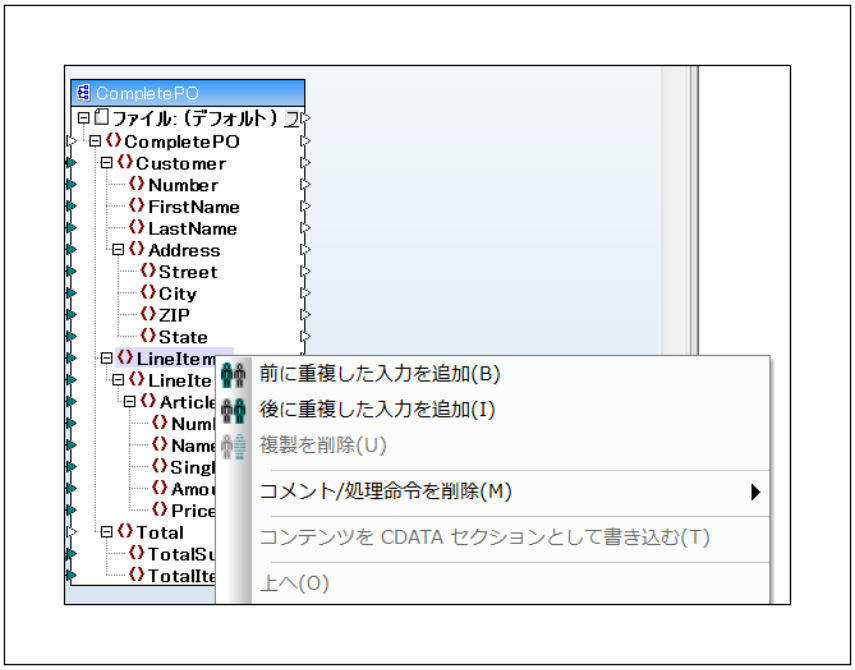
- XML コンポーネント設定
- データベースコンポーネント設定
- CSV コンポーネント設定
- 固定長フィールド コンポーネント設定
- FlexText コンポーネント設定
- JSON コンポーネント設定
- Excel 2007+ コンポーネント設定
- XBRL コンポーネント設定
- EDI コンポーネント設定

XML などの、ファイルベースのコンポーネント、**「ファイル/文字列」**（File/String）ボタンがルートノードの横に表示されます。このボタンは、単一のマッピング内で複数のファイルを処理または生成する際に適用することのできる高度なオプションを指定します（複数の入力または出力ファイルを動的に処理を参照）。更に、このボタンは文字列の解析またはデータを文字列にシリアル化するための高度なオプションを有効化します（文字列の解析とシリアル化を参照）。

4.2.4 入力の複製

1つ以上のソースからデータを受け入れるためにコンポーネントを構成する必要がある場合があります。例えば、2つの異なるスキーマからのデータを単一のスキーマに変換する必要があるとします。目的のスキーマが2つのソーススキーマからのデータを受け入れるようにするには、コンポーネント内の入力アイテムを複製して行います。入力の複製は、ターゲットコンポーネントであるコンポーネントにとってのみ意味があります。与えられたターゲット コンポーネントは、必要に応じて、必要な数複製することができます。

特定の入力アイテムを複製するには、アイテムを右クリックして、コンテキストメニューから**「後 前に入力を複製する」**を選択します。



上のイメージでは、アイテム`LineItem` が 2 番目のソースからマップを可能にするために複製されています。

入力を複製すると、元の入力と複製された入力の間をマップすることができます。例 : これによりデータをソース A からもとの入力に、またデータをソース B から複製された入力にコピーすることができます。

メモ: XML インスタンスを無効にするため、XML 属性の複製は許可されていません。スキーマ内の要素の `maxOccurs` 属性の値に関わらず、XML 要素の場合、入力の複製は許可されています。この振る舞いは、スキーマが後に変更することが可能なため、また、ソースデータが任意のため、意図的です。 例えば、マッピングは、マッピング上で入力が複製されても、単一の XML 要素を生成することができます。

順序を追った例は、複数のソースから1つのターゲットにマップを参照してください。

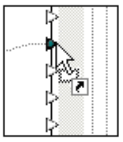
4.3 接続との作業

マッピングは、データを1つのフォーマットから他のフォーマットに変換することを意味します。基本的なマッピングシナリオでは、マッピングエリアにソースとターゲットデータを表すコンポーネントを追加します（例：ソースXML スキーマと目的のスキーマ）。そして視覚的なマッピング接続を2つの構造間に描きます。ですから、接続はソースから目的地へどのようにデータがマップされたかを表す視覚的な表示といえるでしょう。

コンポーネントには、マッピングで小さい三角形として表示される入力と出力があり、これらはコネクタと呼ばれます。出力コネクタは接続を描くアイテムの右横に配置されます。

2つのアイテム間に接続線を引く：

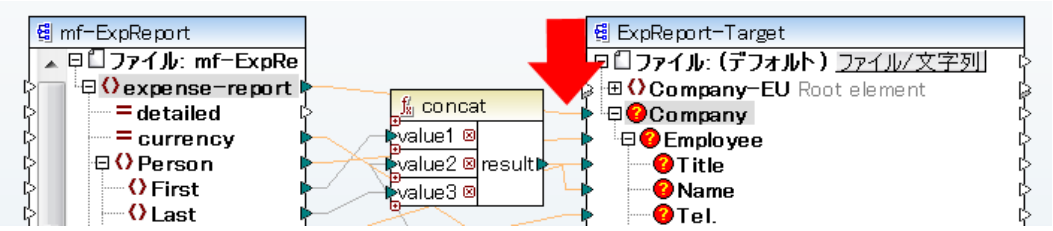
- ソースアイテムの出力コネクタをクリックし、目的アイテムまでドラッグします。ドロップアクションが許可されると、ヒントのリンクがカーソルの横に表示されます。



入力コネクタは入力される接続のみを受け入れます。同じ入力に第2番目の接続を追加使用とすると、接続を新規の接続と置き換えるか、または入力アイテムを複製するかを問うメッセージボックスが表示されます。出力コネクタは複数の接続を持つことができ、それぞれが異なった入力を持ちます。

異なったアイテムに接続を移動する：

- 接続のスタブ（ターゲットに近い直線のセクション）をクリックして、移動先までドラッグします。

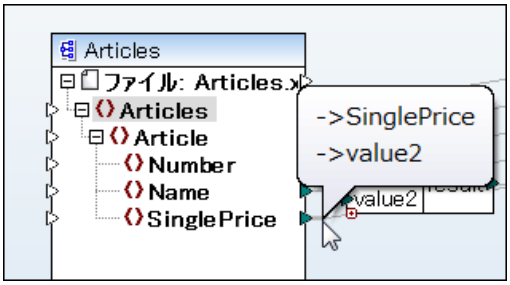


異なったアイテムへの接続をコピーする：

- 接続のスタブ（ターゲットに近い直線のセクション）をクリックして「Ctrl」キーを押しながら移動先までドラッグします。

接続の対極にあるアイテムを表示する：

- マウスポインターを（入力/出力アイコンの近くにある）コネクタの直線部へマウスオーバーすることで、コネクタがハイライトされ、ポップアップが表示されます。ポップアップには、コネクタの対極にあるアイテムの名前が表示されます。同一の出力アイコンから複数の接続が定義されている場合には、最大10の名前が表示されます。スクリーンショットでは、**SinglePrice** と**value2** という複数のターゲットに対して接続が行われていることが理解できます



接続設定 を変更するには以下を行います :

- **「接続」** メニューから **「プロパティ」** をクリックします。 (このメニューアイテムは、接続を選択すると有効化されます)。
- 接続をダブルクリックします。
- 接続を右クリックして、 **「プロパティ」** をクリックします。

接続設定を参照してください。

接続を削除するには、以下を行います :

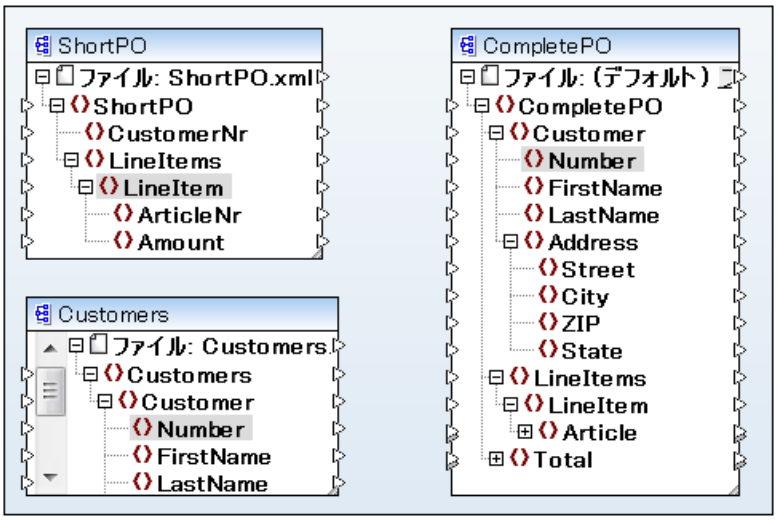
- 接続をクリックして、 **「削除」** キーを押します。
- 接続を右クリックして、 **「削除」** をクリックします。

4.3.1 必須の入力について

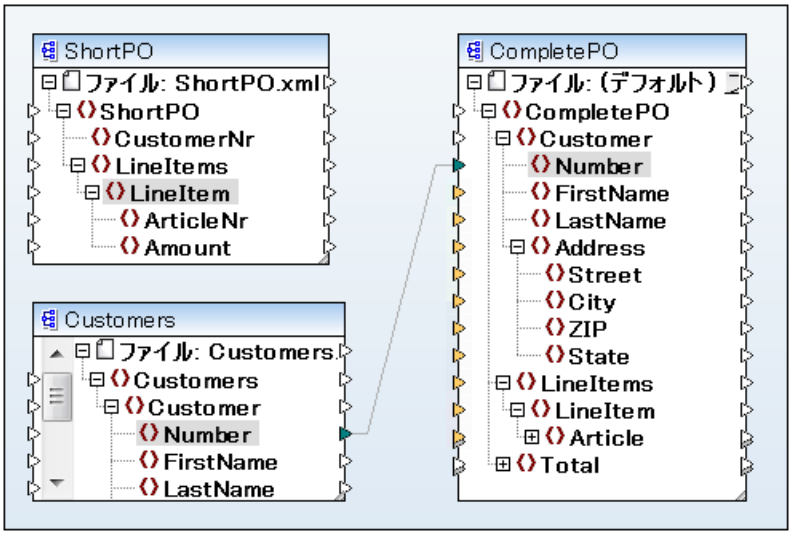
マッピング処理内での手助けをするために、MapForce は、ターゲットコンポーネント内の必須の入力をオレンジ色でハイライトします :

- XML とEDI コンポーネントは、minOccurs パラメーターが 1 と同じまたは 1 より大きいアイテムです。
- In データベース内では、これらは『ゼロではない』として定義されるフィールドです。
- WSDL 呼び出しとWSDL レスポンス (全てのノード)
- 必須と定義されたXBRL ノード
- 関数内では、これらは、パラメーターがマップされると、接続の必要な他の必須のパラメーターがハイライトされる特定の必須のパラメーター です。例 : フィルター入力パラメーターの1つがマップされると、他のパラメータは自動的にハイライトされます。
- MS Excel シート内のワークシート名

例 :
...\MapForceExamples フォルダー内にある、CompletePO.mfd などのマッピング作成の際、挿入されたXML スキーマファイルは下に表示されるとおりです。



Customers コンポーネントのNumber 要素はCompletePO コンポーネントのNumber 要素に接続されています。接続が作成されると、CompletePO コンポーネントの必要なアイテム/ノードは、ハイライトされます。折りたたまれた「Article」ノード/アイコンもハイライトされます。



4.3.2 優先する接続線の表示の変更

MapForce ではマッピングウィンドウに表示されるコネクタの表示方法を選択することができます

選択コンポーネントの接続線の表示 により以下のように表示が切り替わります：

- 全てのマッピングコンポーネントが黒色で表示されるか、
- 現在選択されているコンポーネントに関係したコネクタが黒色で表示されるとともに、その他の接続がグレーで表示されます。

ソースからターゲットへの接続線の表示により以下の表示が切り替わります：

- 現在選択されているコンポーネントに**直接**接続されているコネクタが黒色で表示されるか、
- 現在選択されているコンポーネントの入力または出力アイコンへリンクされているコネクタが黒色で表示されます。

4.3.3 接続の注釈

個別の接続をマッピングに詳細なコメントを与えラベル付けすることができます。 このオプションは **「全ての接続の型」**で使用することができます。

接続に注釈をつける：

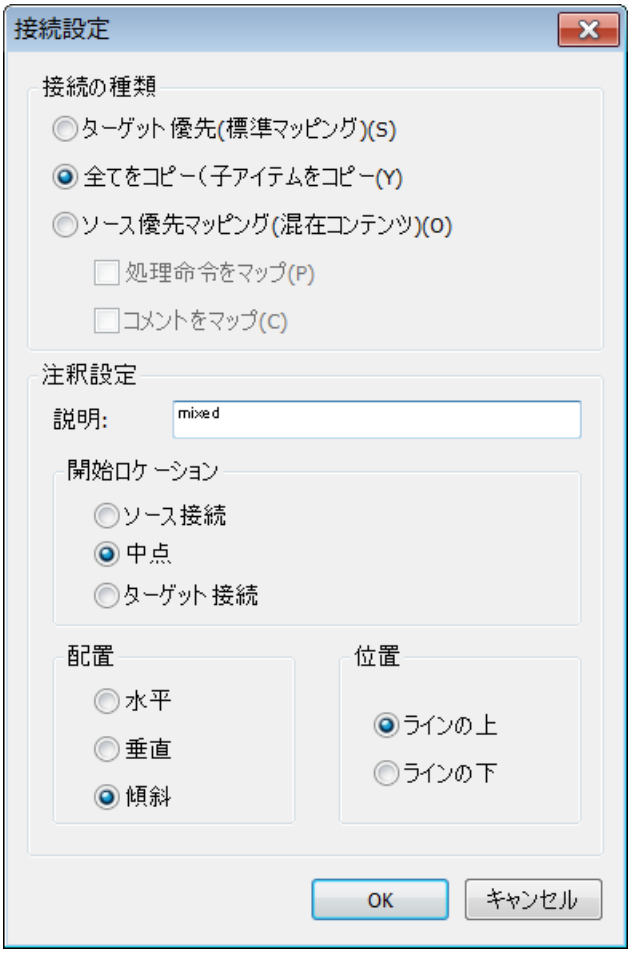
1. 接続を右クリックして、コンテキストメニューからを **「プロパティ」**選択します、
2. 現在選択されている接続の名前 を **「詳細」** フィールドに入力します。これにより注釈設定グループ内の全てのオプションが有効化されます。
3. 残りのグループを使用して、ラベルの **「開始の箇所」、「配置」、「位置」**を定義します。
4. 注釈テキストを確認するには、表示オプションツールバー内の **「注釈の表示」** アイコンを有効化します。



メモ： **「注釈の表示」**アイコンが無効化されている場合、接続にカーソルをポイントすると、注釈を表示することができます。 **「ヒントの表示」** ツールバーボタンが表示オプションツールバー内で有効化されている場合、注釈のテキストは吹き出し内に表示されます。

4.3.4 接続設定

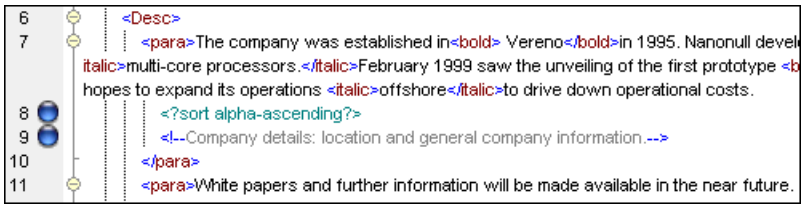
コネクタを右クリックしてコンテキストメニューから**プロパティ**を選択するか、コネクタをダブルクリックして接続設定ダイアログボックスを開き、現在のコネクタに対して特別な（混合コンテンツ）設定を定義することができます。使用することのできないオプションは無効化されています。



接続設定ダイアログボックス

complexType のアイテムのために、以下のマッピングのための接続型を選択することができます（これらの設定は、テキストノードを持たない**complexType** アイテムにも適用することができます）：

ターゲット優先（標準）	接続の型をターゲット優先に変更します（ターゲット優先（標準）マッピング を参照）。
全てコピー（子アイテムのコピー）	接続の種類を「全てコピー」に変更し、ソースとターゲットコンポーネント内の全ての等しいアイテムと自動的に接続します（全てコピー接続 を参照）。
ソース優先（複合コンテンツ）	接続の種類を"ソース優先"に変更し、またマップされる追加要素の選択を有効化します。マッピングに適するために、追加要素はXML ソースファイル内のマップされたアイテムの子アイテムである必要があります。 **「処理命令をマップ」**または**「コメントをマップ」**チェックボックスを有効化することにより、これらのデータグループを出力ファイルに含むことができます。



メモ:CDATA セクションはテキストとして扱われます。

注釈設定グループにより接続へ注釈を与えることができます (接続の注釈 を参照)。

4.3.5 マッチする子要素の接続

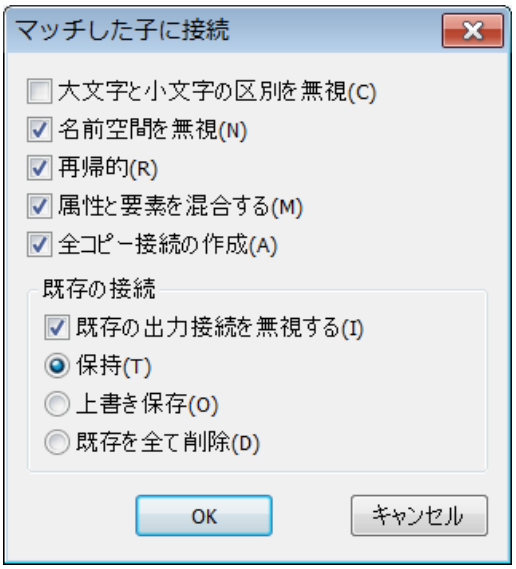
ソースとターゲットコンポーネントの両方内で**同じ名前**のアイテム間で複数の接続を作成することができます。 全てコピー」接続は、デフォルトで作成されます (全てコピー接続を参照)。

「マッチする子を自動的に接続する」オプションをオンまたはオフに切り替えるには、以下を行います:

- **「マッチする子要素の自動接続」**( ) ツールバーボタンをクリックします。
- **接続」**メニューから**「マッチする子要素の自動接続」**をクリックします。

"マッチする子を接続する"のための設定を変更する方法:

1. 両方のコンポーネント内のと **「子アイテム」**いう名前を共有する 2つの 親)アイテムを接続します。
2. 接続を右クリックして、 **「一致する子要素の接続」**オプションを選択します。



3. 必要なオプションを選択して (下のテーブルを参照してください)、 **「OK」**をクリックします。ダイアログボックス内の設定定義に当てはまる、接続が同じ名前の全ての子アイテムのために作成されます。

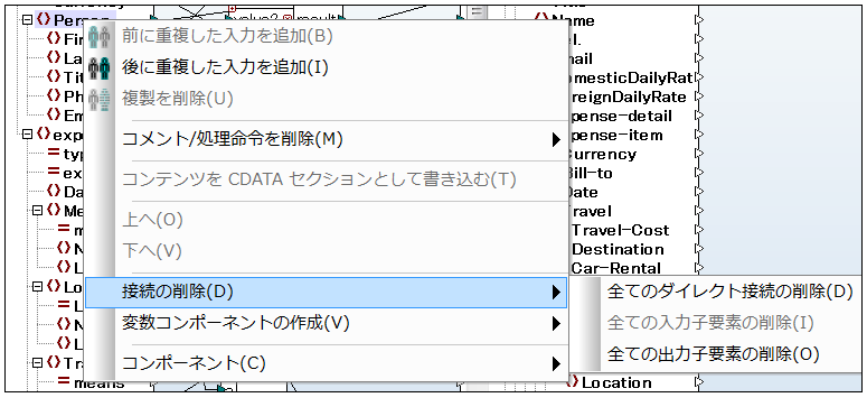
メモ: **「子の自動接続の切り替え」**( ) ツールバーボタンがアクティブな場合、ここで定義する設定が2つのアイテム

を接続する際に適用されます。

大文字と小文字の区別を無視	子アイテム名の大文字と小文字の区別を無視します。
名前空間を無視	子アイテムの名前空間を無視します。
再帰的	一致するアイテム間の接続を再帰的に作成します。すなわち、階層構造内でどれほど深くアイテムがネストされていても、アイテムが同じ名前を持つ限り、アイテム間の接続が作成されます。
属性と要素を混合	有効化されている場合、同じ名前を持つ属性と要素の間の接続を許可します。例：2つの「Name」アイテムが存在する場合、1つが属性、もう1つが要素である場合でも、接続を作成します。
全コピー接続の作成	この設定は、デフォルトの設定では有効化されています。この設定は、「全てコピー」とソースとターゲットアイテム間の接続を（可能であれば）作成します。
既存の出力接続を無視	出力接続が既存の場合でも、追加の接続を一致するアイテムのために作成します。
保持	既存の接続を保持します。
上書きを保存	定義された設定に従い接続を再作成します。既存の接続は破棄されます。
既存を全て削除	新規接続を作成する前に、既存のすべての接続を削除します。

接続の削除

接続：マッチする子を接続するダイアログを使用して作成された接続、または、マッピング処理中の接続をグループ化して削除することができます。



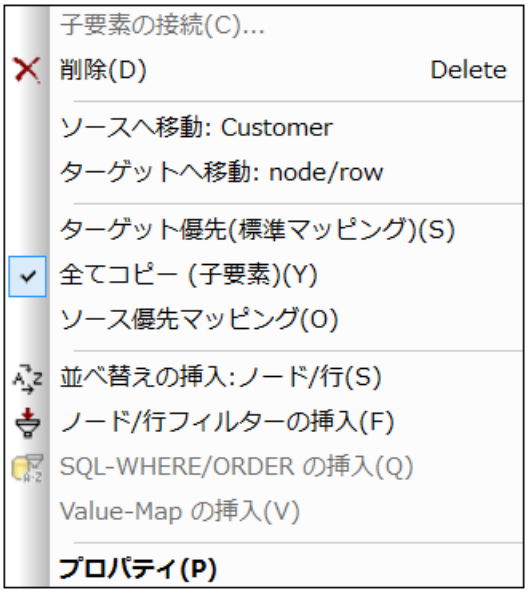
接続の削除：

1. コンポーネント内のアイテムの名前を右クリックします（このサンプルでは「Person」）。
2. **接続の削除 | 全ての..接続の削除」**を選択します。

全てのダイレクト接続を削除	直接マップされた、または現在のコンポーネントから他のソースまたはターゲットコンポーネントにマップされたすべての接続を削除します。
全ての入力子接続を削除	ターゲットコンポーネント内でアイテムを右クリックした場合のみアクティブになります。全ての入力子接続を削除します。
全ての出力子接続を削除	ソースコンポーネント内でアイテムを右クリックした場合のみアクティブになります。全ての出力子接続を削除します。

4.3.6 コンテキストメニューの接続

接続を右クリックすると、以下のコンテキストコマンドを使用することができます。



マッチする子を接続	「マッチする子を接続する」ダイアログボックスを開きます。(マッチする子要素の接続 を参照)。接続がマッチする子アイテムを持つことができる場合、このコマンドを有効化することができます。
削除	選択された接続を削除します。
ソースへ移動 : <item name>	現在の接続のソースコネクタを選択します。
ターゲットに移動 : <item name>	現在の接続のターゲットコネクタを選択します。
ターゲット優先 (標準)	接続の型をターゲット優先に変更します (ターゲット優先接続 を参照)。
全てコピー (子アイテムのコピー)	接続の種類を 全てコピー」に変更し、ソースとターゲットコンポーネント内の全ての等しいアイテムと自動的に接続します (全てコピー接続 を参照)。 ソースアイテムとターゲットアイテムの両方が子アイテムを持つ場合、このコマンド

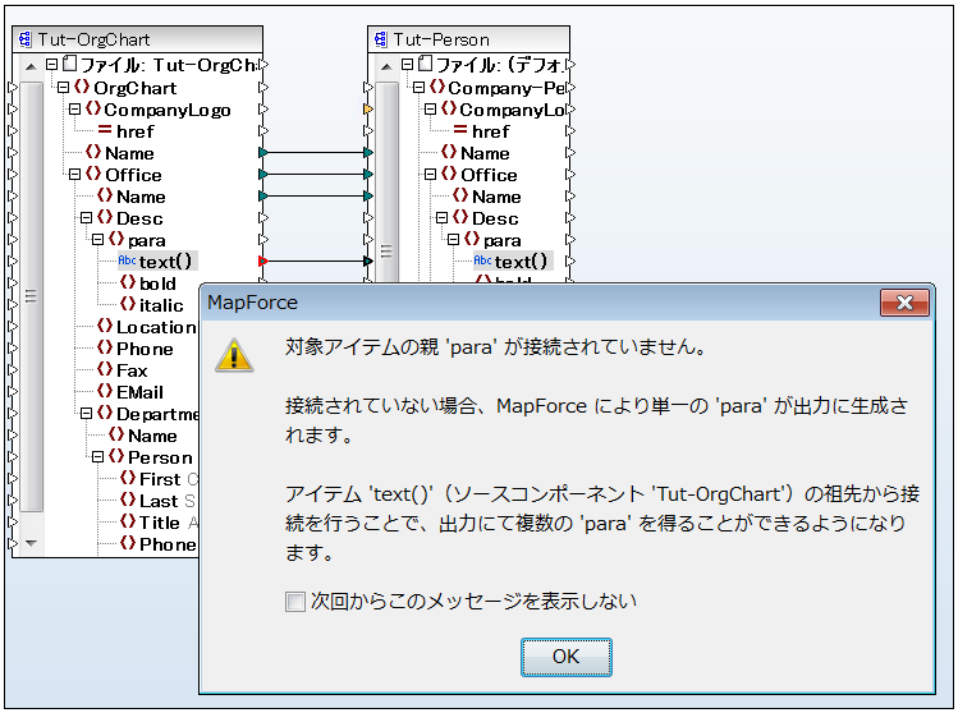
	は有効化されます。
ソース優先（複合型コンテンツ）	接続の種類を「ソース優先」に変更します（ソース優先マッピングを参照）。 ソースアイテムとターゲットアイテムの両方が子アイテムを持つ場合、このコマンドは有効化されます。
並べ替えの挿入：Nodes/Rows	ソースとターゲットアイテム間のソートコンポーネントを追加します。（データの置換えを参照）
フィルターの挿入：Nodes/Rows	ソースとターゲットアイテム間のフィルターコンポーネントを追加します。（フィルターと条件を参照）
SQL-Where 条件の挿入	ソースとターゲットアイテム間の SQL-Where コンポーネントを追加します。（SQL WHERE / ORDER コンポーネントを参照）。
Value-Map の挿入	ソースとターゲットアイテム間の Value-Map コンポーネントを追加します。（Value-Maps の使用 を参照）
プロパティ	「接続設定」ダイアログボックスを開きます。（接続設定 を参照）

4.3.7 見つからない親接続の通知

ソースならびにターゲットアイテム間の接続を手動で作成する場合、起こりえるマッピング結果が MapForce により分析されます。子アイテム同士をマッピングした場合、ソースアイテムの親とターゲットアイテムの親も接続するように促すプロンプトが表示されます。

これはマッピングのプレビューを出力ウィンドウにて行う際に、子アイテムの表示を 1 つに限定しないためのものです。ソースノードから得られるものが単一の値ではなく、シーケンス形式の場合、通常親アイテムの接続を行う必要があります。

<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial フォルダーに収められている **Tut-OrgChart.mfd** マッピングを以下に示します。ソースコンポーネントにある text() アイテムがターゲットコンポーネントにある text() アイテムへ接続されると、メッセージボックスが表示され、親アイテムの para が接続されておらず、出力では一度だけ生成される旨が通知されます。複数の para アイテムをターゲットにて表示するには、ソースならびにターゲットアイテムの para を接続する必要があります。



Tut-OrgChart.mfd (MapForce Basic Edition)

ターゲット内で複数の `para` アイテムを生成するには、ソースとターゲット `para` アイテムをそれぞれ接続します。

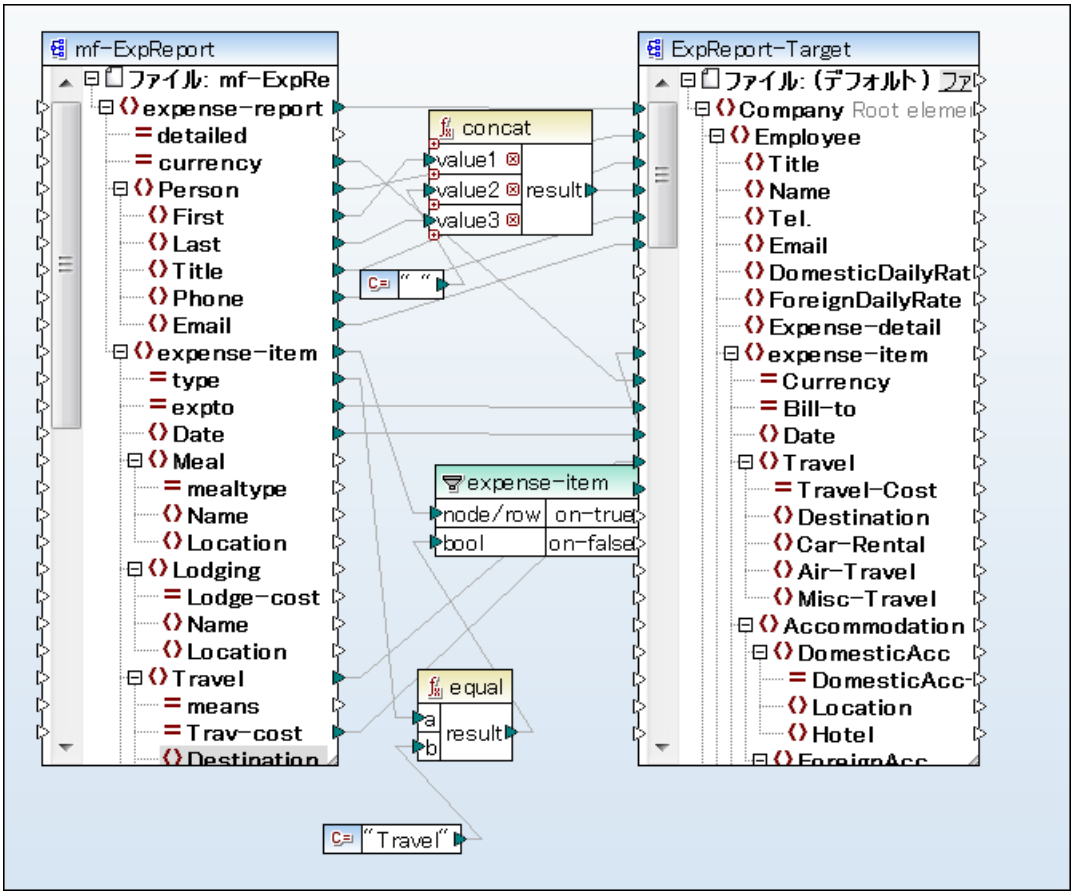
上に表示されるようなメッセージを非表示にするには以下を行います：

1. **「ツール」** メニューから **「オプション」** をクリックします。
2. **「メッセージ」** グループをクリックします。
3. **接続を作成した際、祖先アイテムへの接続を提案アイテム** チェックボックスをクリックします。

4.3.8 接続と子接続の移動

異なるコンポーネントに接続を移動する場合、MapForce は、自動的に同じ子接続をマッチし、この接続が新しい場所に移動されるべきかプロンプトします。この機能の通常の使用方法は、既存のマッピングがある場合、ターゲットスキーマのルート要素の変更です。通常、このような状況が発生すると、すべての降順 接続 を手動でマッピングを再度行う必要がありますが、この機能はそのような状況を回避するために役に立ちます。

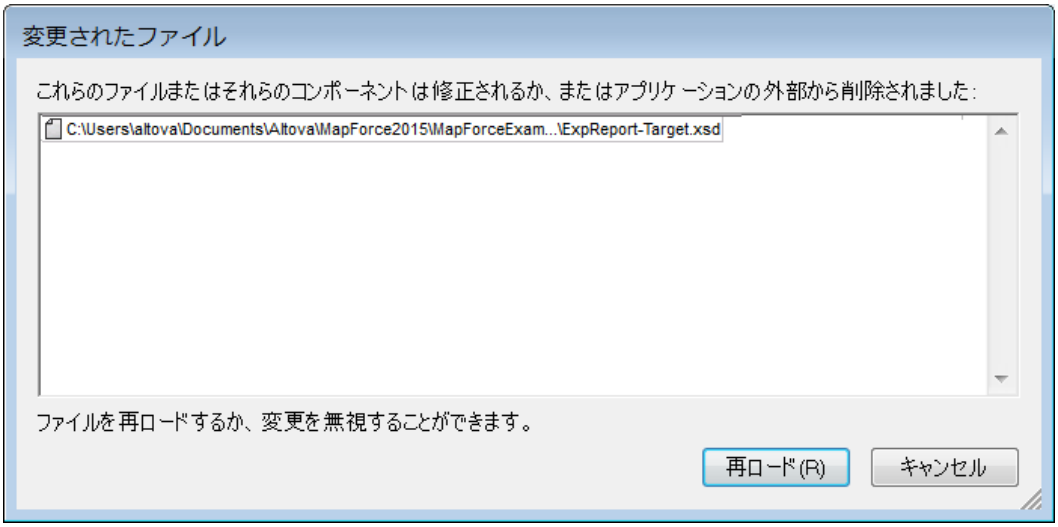
<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial フォルダーで使用することのできる**Tut-ExpReport.mfd** ファイルは、この例で修正され、別名で保存されます。



Tut-ExpReport.mfd (MapForce Basic Edition)

動作を理解するためには、以下を行います：

1. **Tut-ExpReport.mfd** サンプルマッピングを開きます。
2. ターゲットスキーマの`Company` ルート要素を`Company-EU` に変更するために、MapForce の外部で **ExpReport-Target.xsd** スキーマを編集します。MapForce を閉じる必要はありません。
3. ターゲットスキーマの`Company` ルート要素を`Company-EU` に変更すると、MapForce に変更されたファイル"プロンプトが内に表示されます。



4. 更新されたスキーマを再ロードするために、**「再ロード」**ボタンをクリックします。ルート要素が削除されたため、コンポーネントは複数の見つからないノードを表示します。



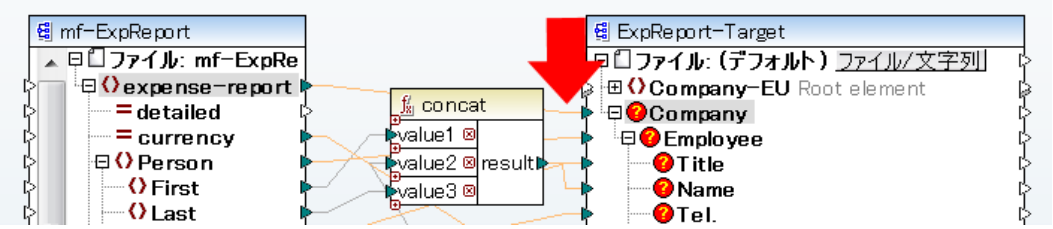
5. コンポーネントの上の **「新しいルート要素の選択」**をクリックします。(ルート要素をコンポーネントヘッダーを右クリックしてコンテキストメニューから **「ルート要素の変更」**を選択することで、ルート要素を変更することができます。)



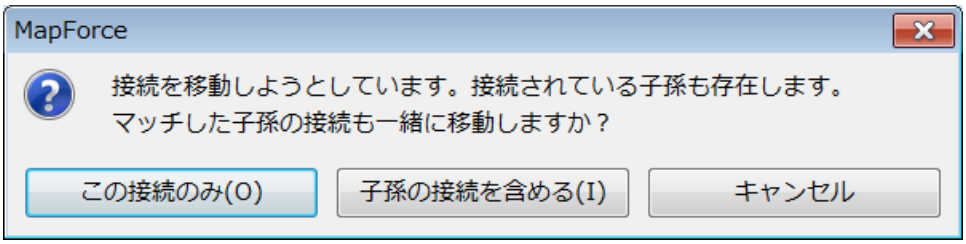
6. `Company-EU` を新しいルート要素として選択し、**「OK」**をクリックして確認します。`Company-EU` ルート要素がコンポーネントの上に表示されます。



7. ソースコンポーネントの`expense-report` アイテムとターゲットコンポーネントの`Company` アイテム間にある接続のターゲットスタブをクリックして、`Company-EU` ターゲットコンポーネントのルート要素の上にドラッグアンドドロップします。



通知ダイアログボックスが現れます。



8. **「子孫の接続を含む」**をクリックする。これにより、新しいルート要素の下で再度マップするようにMapForce に命令し、マッピングは再度有効となります。

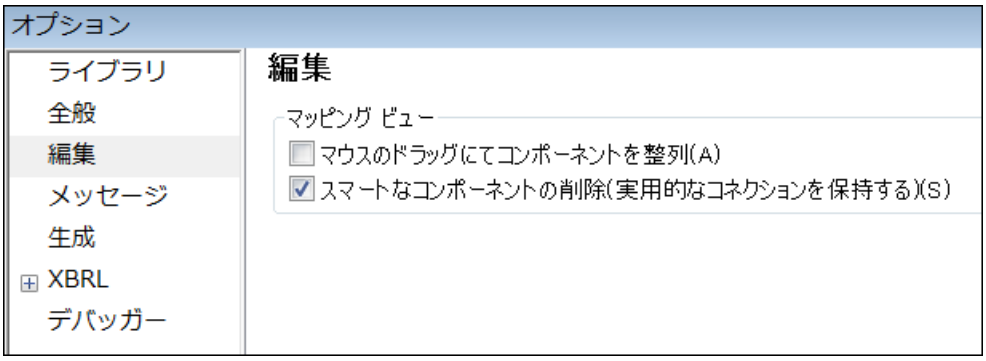
メモ: マッピングするノードが、ソースノードと同じ名前を持ち、異なる名前空間にある場合、通知ダイアログボックスに、以下のを表示する追加ボタンが表示されます: 「子孫とマップ名前空間を含む」。このボタンをクリックすると、同じ名前空間の子接続が同じ子ノードのソース親ノードとして移動されます。このボタンをクリックすると、異なる名前空間ノードの下の同じ子ノードのソース親ノードと同じように同じ名前空間の子接続を移動します。

4.3.9 コンポーネント削除後の接続の保持

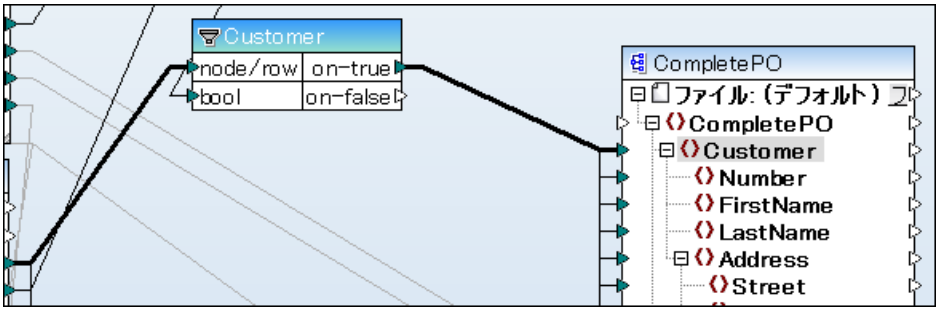
他のコンポーネントに複数の (子) 接続を持つ、コンポーネントを削除すると、何が起こるかを決定することができます。例：フィルター または ソートコンポーネント。全ての子接続を保持し、個別に復元する必要がないためこの機能は役に立ちます。

コンポーネントが削除された後に、子接続を保持または復元すること、または、全ての子接続をすぐに接続することが選択できます。

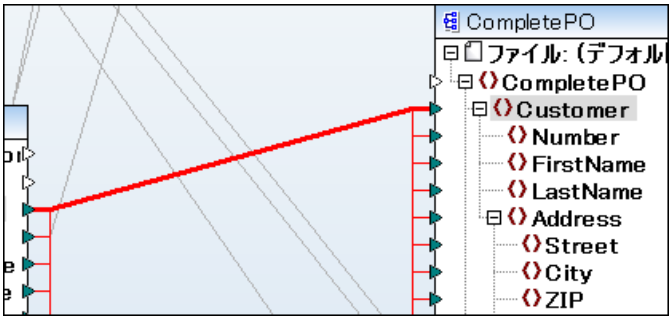
現在の設定を確認するために (タブ) を選択します。チェックボックスのデフォルト設定は、**非アクティブ**です。すなわち「スマートなコンポーネント削除 (役に立つ接続を保持)」が無効化されます。



例：..\MapForceExamples フォルダー内でCompletePO.mfd マッピングを使用するには、チェックボックスがアクティブ化されていると、Customer フィルターは、以下に表示される子アイテムに接続された**全てコピー**接続になります。



Customer フィルターを削除すると、削除の確認を問うプロンプトが表示されます。「はい」を選択すると、フィルターは削除されますが、子接続は保持されます。



その他の接続はまだ選択されています (すなわち、赤で表示される箇所)。削除する場合は、Del キーを押します。

マッピングエリアをクリックすると、コネクタの選択が解除されます。

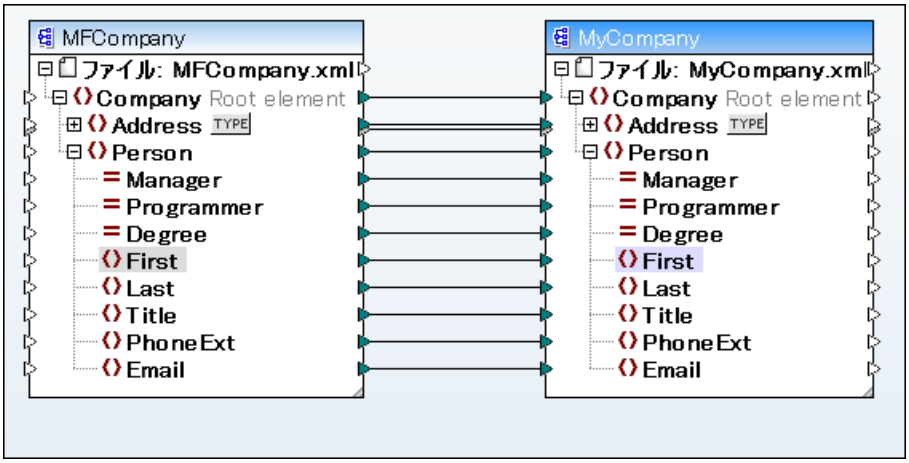
「スマートなコンポーネント削除 ..」チェックボックスが**無効化されている**場合、フィルターを削除すると全ての子接続がすぐに削除されます。

メモ: フィルターコンポーネントが「on-true」と「on-false」出力に接続されている場合、両方の出力のための接続は保持されます。

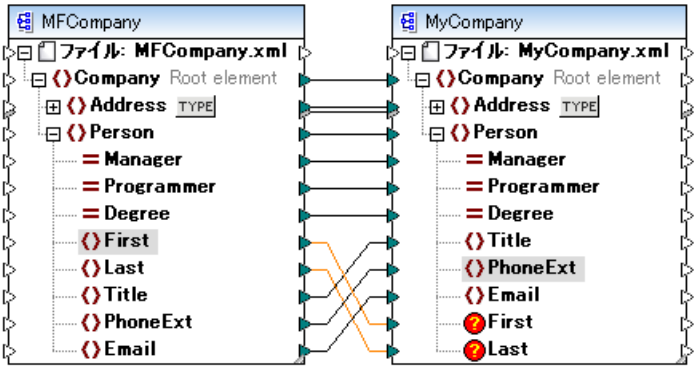
4.3.10 見つからないアイテムの扱い方

時が経つに従って、XML スキーマにて定義された要素や属性が追加/削除され、マッピング内にあるコンポーネントの構造が変更されるということがあります。MapForce ではプレースホルダーアイテムを使用することで、全ての接続を保持し、アイテムが削除された際にコンポーネント間で関連する接続データの保持を行います。

例 :
MFCompany.xsd スキーマファイルをサンプルファイルとして使用します。スキーマファイルを MyCompany.xsd という名前でコピーし、両スキーマ間にて Company アイテムに対する接続を作成します。これで子要素の自動接続が有効になっている場合、コンポーネント間にある全ての子アイテムに対する接続が作成されます。



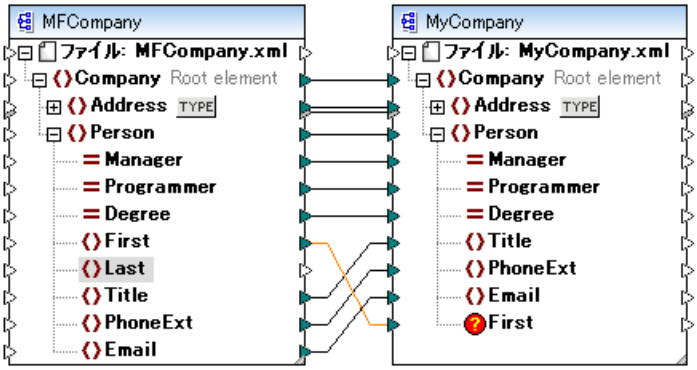
XMLSpy にてMyCompany.xsd を編集し、スキーマ内にあるFirst ならびにLast アイテムを削除します。MapForce へ再度切り替えると、スキーマを**再ロード**するよう促されます。再ロードすることで、MapForce 内にあるコンポーネントが更新されます。



削除された**アイテム**ならびに**コネクタ**が、アイコンとともにMyCompany コンポーネントに表示されます。必要な場合は他のアイテムへコネクタを再接続することができるほか、コネクタを削除することもできます。

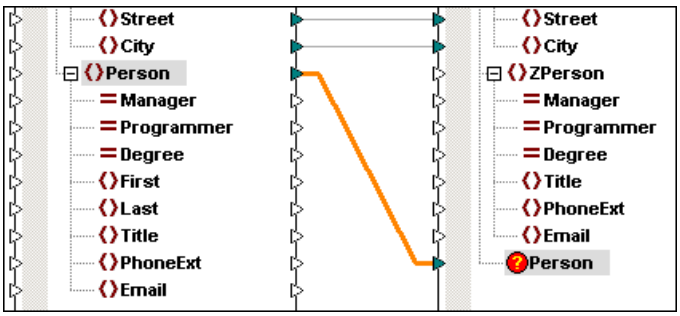
この状態でマッピング (または生成されたコード) のプレビューを行うこともできますが、警告がメッセージウィンドウに表示されます。プレビューならびにコード生成において、上のアイコンで示されるような、見つからないアイテムは無視されます。

ハイライトされているコネクタをクリックし、削除することで、見つからないアイテム (この場合はMyCompany 内部のLast) がコンポーネントから削除されます。



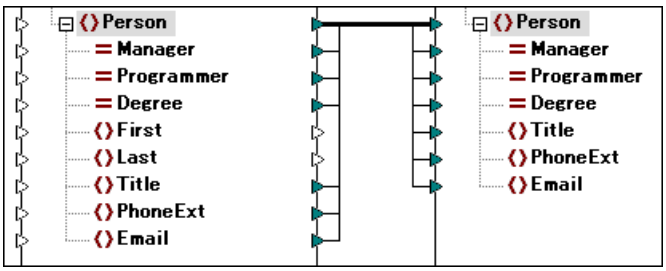
名前変更されたアイテム

例えばPerson という親アイテムがZPerson という名前に変更された場合、オリジナル (Person) の親アイテムコネクタが保持され、子アイテムのコネクタが削除されます。



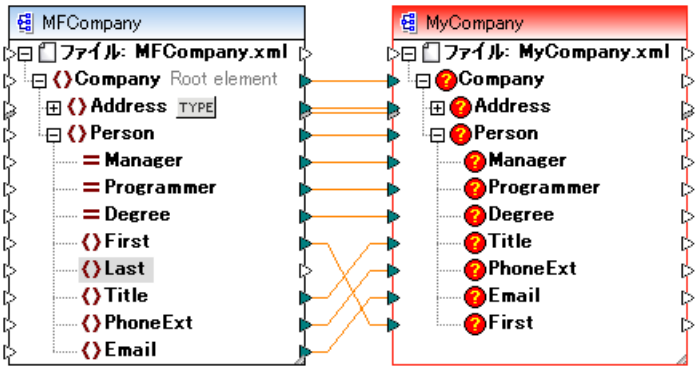
「全てコピー」接続と見つからないアイテム

全てコピー接続は通常の接続と同じように扱われますが、見つからないアイテムへの接続は保持も表示もされません。

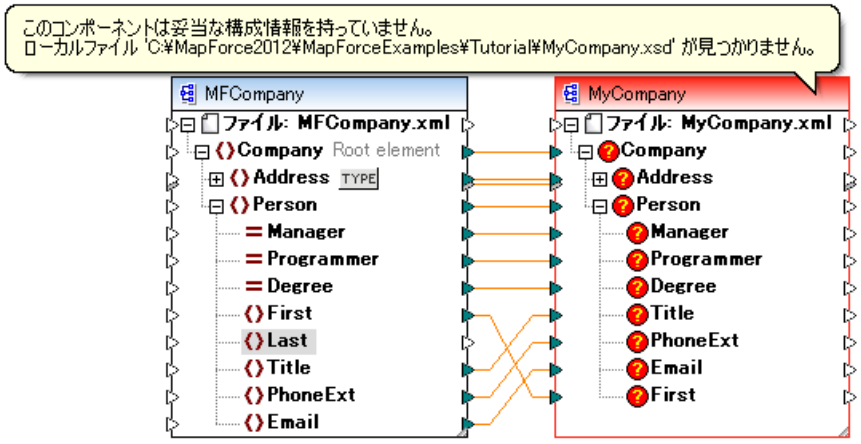


名前変更または削除されたコンポーネントソース

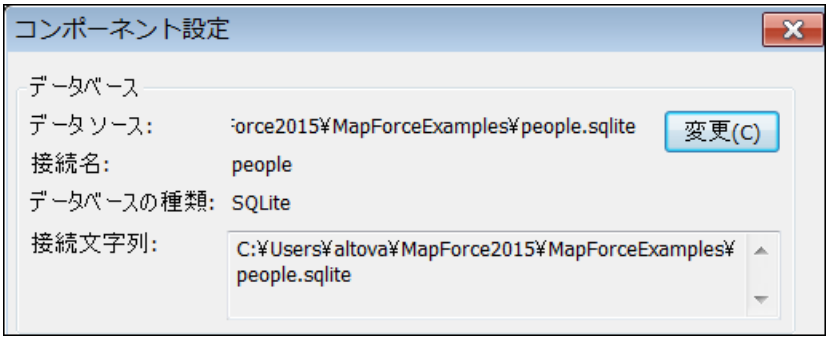
スキーマや、データベースなどのコンポーネントの**データソース**が名前変更または削除された場合、コンポーネントに含まれる全てのアイテムがハイライトされます。コンポーネントの外周であるフレームにより、スキーマまたはデータベースファイルに対して正常な接続がなされていないことが示されます。この状態でプレビューやコード生成を行うことはできません。



マウスカーソルをハイライトされているコンポーネントへマウスオーバーすることで、ポップアップが表示されます。



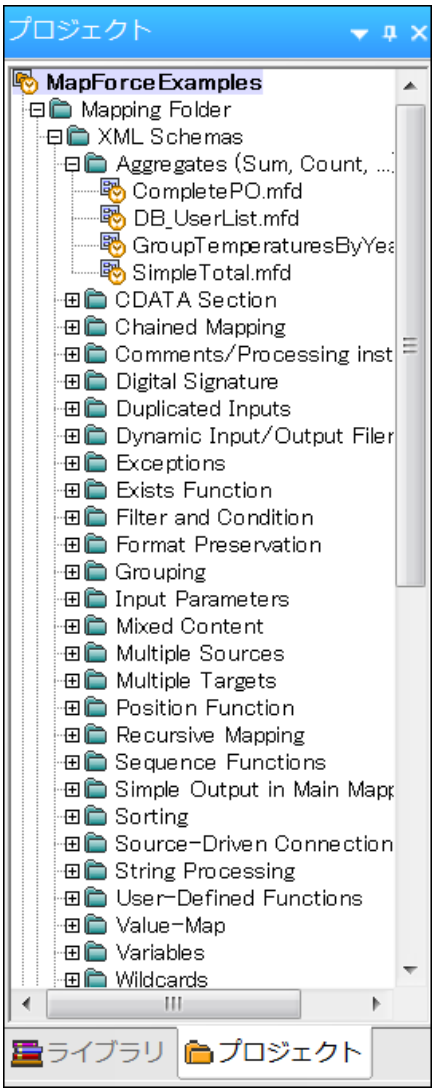
ハイライトされたコンポーネントのタイトルバーをダブルクリックすることで、コンポーネント設定ダイアログボックスが表示されます。**スキーマファイル**のグループにある参照ボタンをクリックすることで別の (またはバックアップとなる) スキーマを選択することができます。詳細についてはリファレンスセクションの "コンポーネント" セクションを参照ください。
コンポーネントがデータベースの場合、ダイアログボックス内の**変更**ボタンをクリックすることで、別のデータベースやテーブルを選択することができるようになります。同名のテーブルへの接続は保持されます。



全ての妥当な (正確な) 接続 (同様の構造を持ったスキーマまたはデータベースを選択した場合全ての妥当な (正確な) 接続が保持されます。

4.4 マッピングプロジェクトとの作業

スタンドアロンのマッピングの作成に追加して、複数のマッピングを含むマッピングプロジェクトも作成することができます。プロジェクトに追加されるマッピングはプロジェクトウィンドウから簡単にアクセスすることができます。



プロジェクト ウィンドウ (MapForce Enterprise Edition)

プロジェクトの主な利点は、特別なプロジェクトに含まれる全てのマッピングのための（ターゲット言語と出力ディレクトリなど）共通のコード生成設定を定義が可能な点です。またプロジェクト内にフォルダーを作成することができ、プロジェクト内の個別のフォルダーのためのカスタムコード生成（C++、C#、とJava）の設定を指定することも可能です。 MapForce-により生成されたプログラムコードに関する詳細は、 コードジェネレータを参照。

MapForce Enterprise エディションでは、Web サービスプロジェクトを追加して作成することができます。このようなプロジェクトは、既存のWeb サービス記述言語（WSDL）ファイルをベースにした、SOAP Web サービスを実装するJava またはC# プログラムコードを生成することを可能にします。Web サービスプロジェクトに関する詳細は、Web サービスの実装を参照してください。

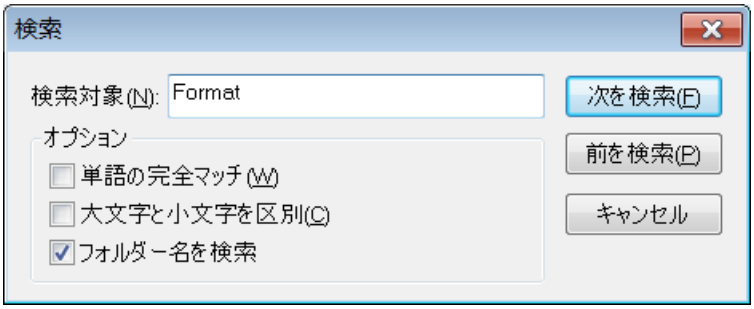
4.4.1 プロジェクトを開く、検索する、閉じる

MapForce プロジェクトファイルは *mfp 拡張子を持ちます。既存のMapForce プロジェクトを（**「ファイル」**メニューから **開く**をクリックして）マッピングと同様に開きます。

MapForce でマッピングプロジェクトが開かれた場合、プロジェクトウィンドウは、プロジェクトに追加された全てのファイルとフォルダーを表示します。デフォルトでは、MapForce をはじめて実行した場合、プロジェクトウィンドウ内で **MapForceExamples.mfp** プロジェクトをロードします。

プロジェクト内でファイルを検索する方法：

1. プロジェクトウィンドウで、検索するプロジェクトまたはフォルダーをクリックします。
2. **「CTRL + F」**を押します。
3. オプションで、検索オプションを選択します。例：検索にフォルダー名を含むには、**「フォルダー名から検索」**オプションを選択します。



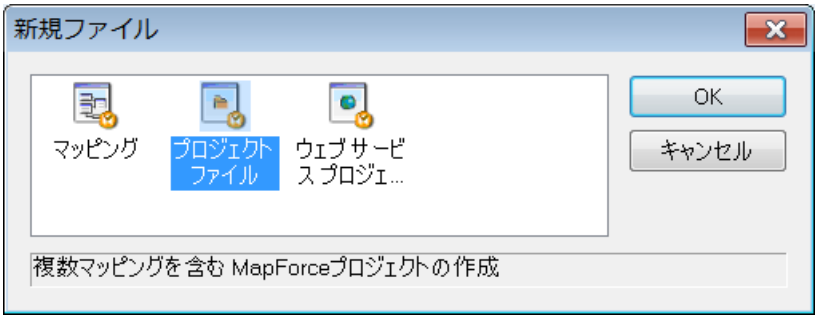
プロジェクトを閉じる方法：

- **「プロジェクト」**メニューから **「プロジェクトを閉じる」**をクリックします。

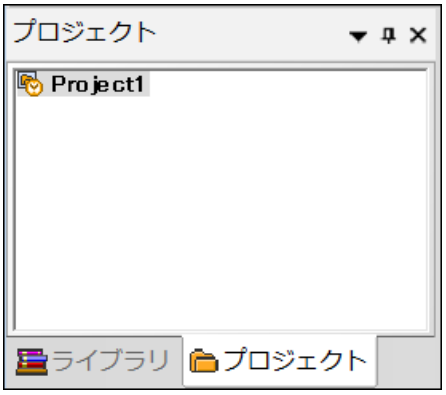
4.4.2 新規プロジェクトの作成

新規プロジェクトを作成する

1. メニューから「ファイル| 新規作成」を選択します。
2. プロジェクトファイルアイコンを選択して、**「OK」**をクリックします。



3. **「プロジェクトに名前をつけて保存」**ダイアログボックスにてプロジェクト名を入力し、**保存」**をクリックします。
4. プロジェクトフォルダーがプロジェクトタブに追加されます。



プロジェクトにマッピングを追加します。

現在アクティブなマッピングをプロジェクトに追加するには、以下を行います：

- **「プロジェクト」**メニューで **「プロジェクトにアクティブファイルを追加する」** をクリックします
- プロジェクトを右クリックして、 **「プロジェクトにアクティブファイルを追加する」**を選択します。

プロジェクトに既存のマッピングファイルを追加するには、以下を行います：

- **「プロジェクト」**メニューで **「プロジェクトにファイルを追加する」**をクリックします。
- プロジェクトを右クリックして、 **「プロジェクトにファイルを追加する」** を選択します。

ヒント：複数のファイルを開くには、開くダイアログボックス内でファイルを選択するときに **「Ctrl」** キーを押します。

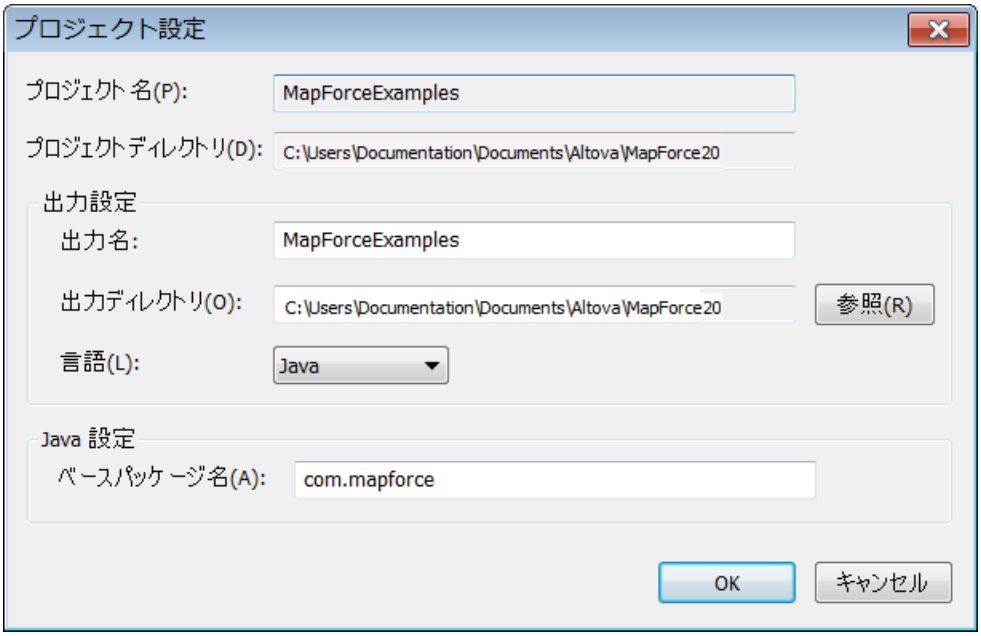
プロジェクトからファイルまたはフォルダーを削除するには、以下を行います：

- プロジェクトウィンドウ内のファイルを右クリックし、コンテキストメニューから、 **「削除」**を選択します。
- プロジェクトウィンドウ内のファイルを選択して、 **「削除」**を押します。

4.4.3 コード生成設定の設定

全てのプロジェクトに対して、プロジェクト内の全てのマッピングに影響するコード生成設定を指定することができます。 **「プロジェクト設定」**ダイアログボックスを開くには、以下を行います：

- **「プロジェクト」**ウィンドウ内のプロジェクト名を右クリックして、 「コンテキスト」メニューから **「プロパティ」** を選択します。
- **「プロジェクト」**メニューから **「プロパティ」**をクリックします。



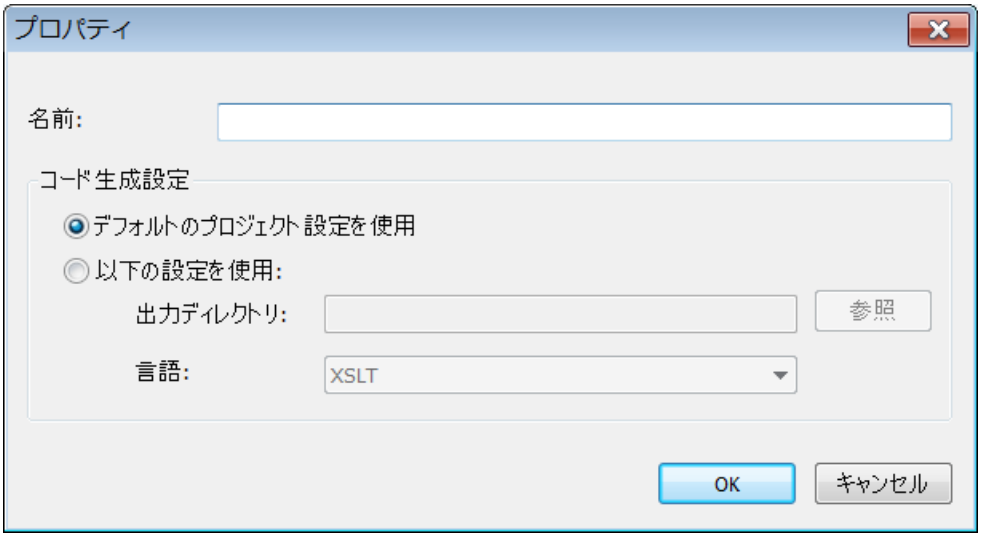
プロジェクト設定ダイアログボックス

使用することのできる設定は、以下のとおりです。プロジェクト名とプロジェクトディレクトリはプロジェクト作成後は変更することができません。

出力名	入力された値は生成されるプロジェクトまたはソリューションの名前、また生成されたコード内のオブジェクトの名前を決定します。
出力ディレクトリ	(このプロジェクト内の全てのマッピングから)生成されたコードが保存される Windows フォルダーを決定します。デフォルトでは、出力はプロジェクトディレクトリ内の **出力** ディレクトリに保存されます。
言語	このプロジェクト内のすべてのマッピングファイルのためのコード生成言語を定義します。
ベースパッケージ名	この設定は、Java を変換言語として選択する場合適用することができます。Java プロジェクト内で生成されたベースパッケージ名を定義します。

4.4.4 プロジェクトフォルダーの管理

プロジェクト内のマッピングをフォルダーに整理する場合、必要に応じてフォルダーを作成することができ、マッピングに追加 (マッピングにドラッグ)することができます。これらのフォルダーは「仮想」であり、MapForce プロジェクト内でのみ意味があります、すなわち、オペレーティングシステム上の実際のフォルダーには対応していません。フォルダーを作成する利点の一つは、その特定のフォルダーのある全てのマッピングファイルに対して共通のコード生成設定 (ターゲット言語と出力ディレクトリなど)を定義することができることです。



フォルダープロパティダイアログボックス

MapForce プロジェクト内にフォルダを作成する:

1. 以下を行います:
   - o **「プロジェクト」** メニューから **「フォルダーの作成」** をクリックします。
   - o プロジェクトを右クリックして、 **「フォルダーの作成」** を選択します
2. プロパティダイアログボックスで、必要とされるコード生成設定を入力して、 **「OK」** をクリックします。

フォルダープロパティダイアログボックス内で定義することのできる設定は、以下のとおりです。

名前	フォルダーの名前
デフォルトプロジェクト設定を使用	これはデフォルトのオプションで現在のフォルダーのコード生成設定はプロジェクトと同じ設定であることを意味します。ですから、プロジェクトからコードを生成する場合、MapForce は、フォルダーレベルではない、プロジェクトレベルで定義されたコード生成設定を使用します。 (プロジェクトレベルで設定されている以外の場合で)フォルダーがカスタムコード生成設定を必要とする場合、 **「以下の設定の使用」** を選択して、コード出力ディレクトリと言語を必要に応じて指定してください。
出力ディレクトリ	(このフォルダー内の全てのマッピングで)生成されたコードが保存される Windows フォルダーを定義します。
言語	このフォルダー内の全てのマッピングファイルのためのコード生成言語を定義します。

Chapter 5

マッピングのデザイン

5 マッピングのデザイン

このセクションは、データマッピングのデザイン方法とマッピングエリア上のデータの変換方法について説明します。また、マッピングデザインに関連した、注意点に関しても説明します。以下のロードマップを利用して、特定のタスクまたはコンセプトに素早くアクセスすることができます:

...を実行するには	以下を参照してください...
その他のスキーマのパス参照とマッピングで使用されるインスタンスと他のファイルの作成と編集を行います。	相対と絶対パスの使用
必要に応じてデータマッピングを微調整します。(例:ターゲットコンポーネント内のアイテムのシーケンスの調整)。	マッピング接続の種類
異なるスキーマを持つ複数のソースからのデータを単一のスキーマにマップします。	複数のスキーマからデータをマージする
コンポーネントの出力を他のコンポーネントの入力として使用します。	チェーンマッピング/パススルーコンポーネント
複数のファイルを同じマッピングにソースまたはターゲットとして処理します。(例:ディレクトリ内の全てのファイル)。	複数の入力または出力ファイルを動的に処理
(文字列パラメーターなど)外部の値をマッピングにパスします。	マッピングにパラメーターを与える
ファイルの代わりに、マッピングから文字列の値を取得します。	マッピングから文字列の値を返す
マッピングデータの一部を後の処理のために一時的に保管します(プログラミング言語内の変数に類似)。	変数の使用
昇順または降順の順序でデータを並べ替える。	データの置換え
特定の条件に従いnodes/rowsをフィルター、または値を処理します。	フィルターと条件
キーと値のペアの処理を行います。例:数値の表示を月数に変換します。(01、02、などを)テキストの表示(一月、二月、などに変換します。	Value-Maps の使用
特定の条件が発生した場合、エラーを返すようにマッピングを構成します。	例外の追加
複雑なマッピング構造を文字列データ型に変換、または、文字列データ型を複雑なマッピング構造に変換します。	文字列の解析とシリアル化
複雑なマッピングのデザインで予期しない結果の発生を回避します。	マッピングのルールと戦略

重要な点は、MapForce は、多種の処理タスクを助ける、広範囲なビルドイン関数ライブラリを搭載している点です(関数ライブラリリファレンスを参照)。ビルドインライブラリでも、十分ではない場合、自身のカスタム関数を MapForce で作成し、外部 XSLT ファイルおよび dll または Java クラスライブラリで再利用することができます。更に詳しい情報に関しては、関数の使用を参照してください。

5.1 相対と絶対パスの使用

マッピングデザインファイル (*.mfd) は、複数のスキーマとインスタンスファイルを参照する場合があります。MapForce により使用されるスキーマファイルは、マップされるデータの構造を決定し検証します。インスタンスファイルは他方、スキーマに対してソースデータを読み込み、プレビューし、検証することが要求されます。

マッピングは PDF、HTML とWord などの出力のためにデータをフォーマットするために StyleVision Power Stylesheets (*.sps) ファイルへの参照を含む場合があります。また、マッピングは Microsoft Access または SQLite などのファイルベースのデータベースへの参照を含む場合があります。

マッピングにコンポーネントを追加する場合、マッピングデザインで使用されるファイルへの参照はすべて MapForce により作成されます。ですが、上記のパス参照を常に手動で設定または変更することができます。

このセクションは設定のための命令、またはマッピングにより参照されるその他のファイル型にパスの変更方法、相対と絶対パスの使用が指する意味について説明します。

5.1.1 コンポーネント上で相対パスを使用する

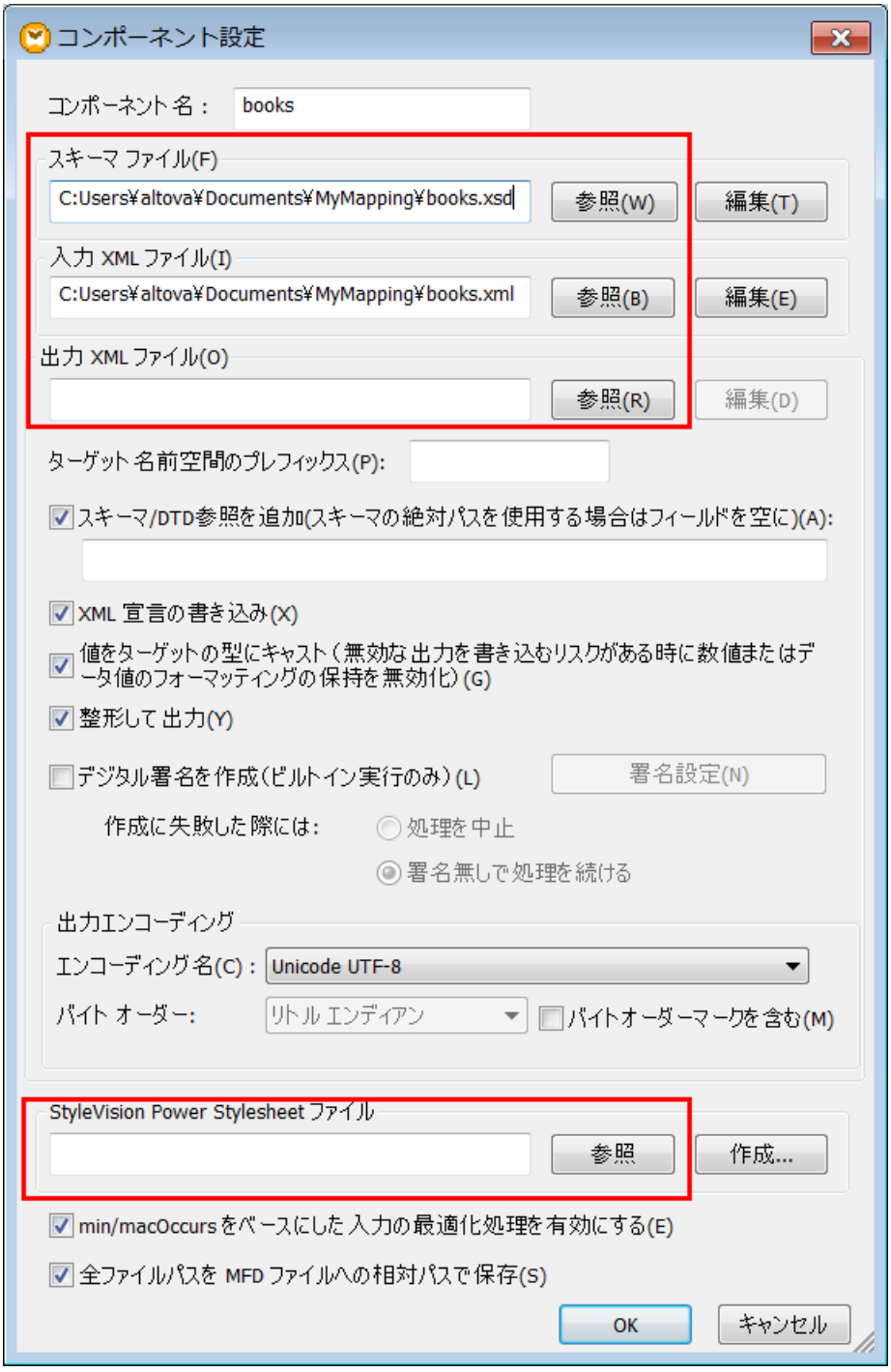
コンポーネント設定ダイアログボックス (XML コンポーネントのために下に表示されています) は、コンポーネントにより参照される可能性のある複数のファイルのための絶対または相対パスを指定するオプションを提供します:

- 入力ファイル (すなわち、MapForce がデータを読み込むファイル)
- 出力ファイル (すなわち、MapForce がデータを書き込むファイル)
- スキーマファイル (スキーマを持つコンポーネントに適用可能)
- 構造ファイル (ユーザー定義関数の入力または出力パラメーター, または変数などの複雑な構造を持つコンポーネントに適用可能)
- PDF、HTML とWord などの出力のためのデータをフォーマットするために使用される、StyleVision Power Stylesheet (*.sps) ファイル

対応するテキストボックス内で相対パスを直接入力することができます (下のイメージで赤線の枠内に表示されています)。

> 相対ファイルパスを入力する前に、マッピングファイル (.mfd) を最初に保存してください。それ以外の場合、全ての相対パスはのWindows のパーソナルアプリケーションフォルダーに対して解決されます (ドキュメント\Altova\MapForce2017) これは、予期しない振る舞いの場合があります。

.mfd ファイルマッピングに対して相対的な上記のファイルパスすべてを保存するようにMapForce に命令することができます。下のサンプルイメージでは、**「MFD ファイルに相対的なすべてのファイルパスを保存する」**オプションに注意してください。(デフォルトかつ奨励されるオプションである) チェックボックスが有効化されていると、コンポーネントにより参照されるファイルのパスは、マッピングデザインファイル (.mfd) のパスに相対的に保存されます。これはコンポーネントに参照される全てのファイルに影響を及ぼします (イメージで赤線の枠内に表示されています)。



コンポーネント設定ダイアログボックス

上で説明されているコンポーネントはXML コンポーネンですが、 設定 **「MFD ファイルに相対的なすべてのファイルパスを保存する」** は、下のファイルでも同じように動作します：

- 複雑なユーザー定義関数の入力または出力パラメーターと複雑な型の変数により使用される構造ファイル
- 入力または出力 フラットファイル *

- XML フィールドをサポートするデータベースコンポーネントにより参照されるスキーマファイル *
- 入力または出力 XBRL、FlexText、EDI、Excel 2007+、JSON ファイル **

* MapForce Professional とEnterprise Edition
** MapForce Enterprise Edition のみ

上のコンポーネントを例として、 mfd ファイルが**books.xsd** と**books.xml** ファイルと同じフォルダーに存在する場合、パスは、以下のとおり変更されます :

C:\Users\altova\Documents\MyMapping\books.xsd は、**books.xsd** に変更されます。
C:\Users\altova\Documents\MyMapping\books.xml は、**books.xml** に変更されます。

ローカルではないドライブまたはURL を使用するパスは相対的にはなりません。

「上書き保存」メニューコマンドを使用して、マッピングを新規のフォルダーに保存する場合、チェックボックスが有効化されている場合、MapForce は、コンポーネントにより参照されるファイルを記録します。また、全てのファイルがマッピングと同じディレクトリに存在する場合、ディスク上の新しい場所にディレクトリ全体を移動してもパスの参照が壊されることはありません。

相対パスを使用することと(**「MFD ファイルに相対的なすべてのファイルパスを保存する」**チェックボックスを有効化)は多くの場合とても重要です。例 :

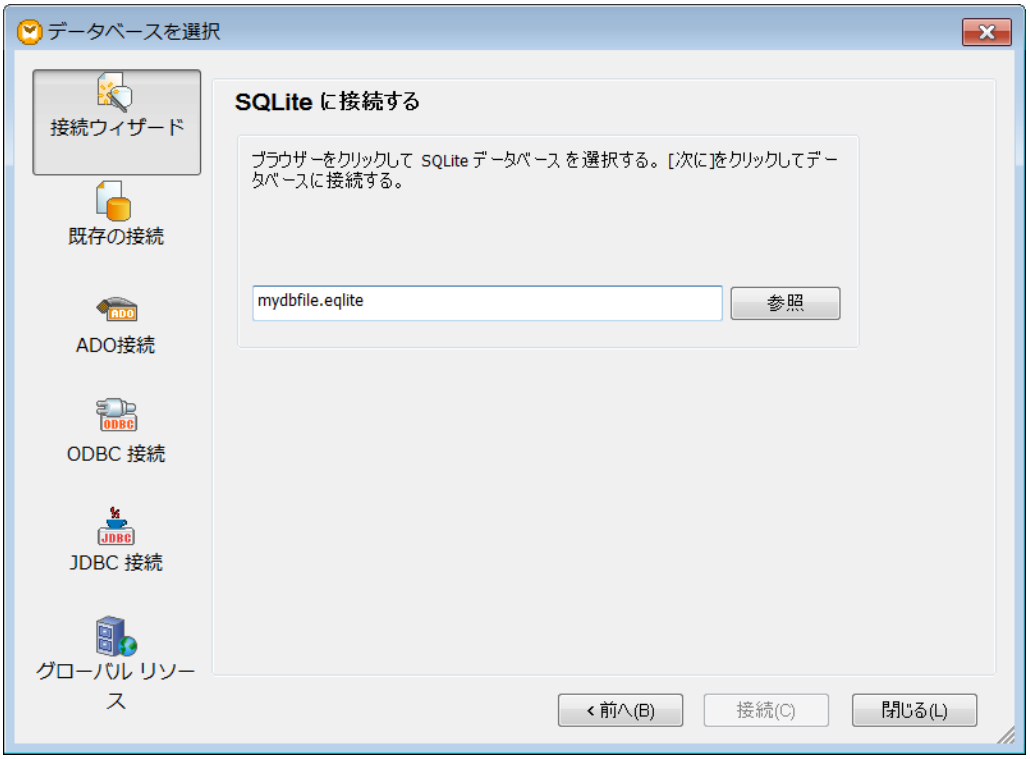
- オペレーティングシステム上のマッピングのロケーションが将来変更される可能性のある場合。
- (TortoiseSVN, などのバージョンコントロールシステムを使用して)マッピングがソースコントロール化にあるディレクトリに保管されている場合。
- 異なったマシンに、または異なったオペレーティングシステムにマッピングを実行のためにデプロイする場合。

「MFD ファイルに相対的なすべてのファイルパスを保存する」チェックボックスが無効化されている場合、マッピングの保存は、ファイルパスを変更しません (すなわち、コンポーネント設定ダイアログボックス内で表示されるままになります)。

5.1.2 ファイルベースのデータベースへパスを設定する

Microsoft Access またはSQLite マッピングなどのデータベースファイルを追加する場合、 (データベース接続ウィザードの開始 を参照)、絶対パスの代わりに相対パスを使用します。相対パスを使用するには、 「データベース接続ウィザード」内の **参照」**をクリックする代わりに必要な相対パスを入力します。

相対ファイルパスを入力する前に、マッピングファイル (.mfd) を最初に保存してください。それ以外の場合、全ての相対パスはのWindows のパーソナルアプリケーションフォルダーに対して解決されます (ドキュメント\Altova \MapForce2017) これは、予期しない振る舞いの場合があります。



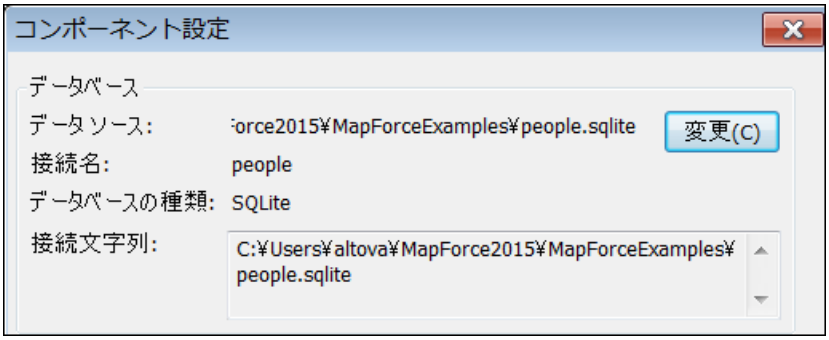
データベース接続ウィザード

データベースがSQLite データベースの場合、以下の条件がtrue の場合、 **「接続」**ボタンが有効化されます：

- マッピング (mfd) ファイルに対して相対的に解決されることのできるファイルをパスはポイントしています。
- 参照されるファイルはSQLite データベースです。

マッピング内に既存のデータベースのパスコンポーネントを変更するには、以下を行います：

1. データベースコンポーネントのヘッダーを右クリックして、 **「プロパティ」**（ コンポーネント設定を変更するも参照 ）を選択します。または、 コンポーネント タイトルバーをダブルクリックします。
2. コンポーネント設定ダイアログボックスから、 **「変更」**をクリックします。



これにより、(パスを含む)データベース接続プロパティを変更することのできる、データベース接続ウィザードが再度開かれます。

"接続文字列"は、常に絶対パスを含むことに注意してください。コンポーネント内の構造情報のためにはデータベースが使用されています。"データソース"内の相対パスは、コンポーネントが相対ファイルパスと共に作成されたことを示します。

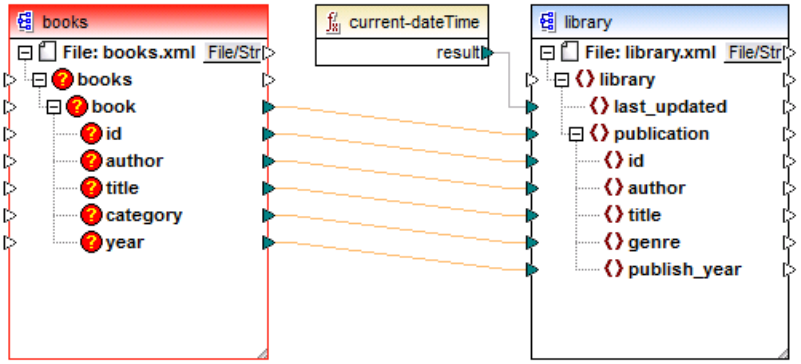
コードを生成する場合、または、MapForce Server 実行ファイル (mfx) をコンパイルする場合、データベースのパスは、マッピング設定に従い (生成されたコード内のパスについてを参照) 絶対または相対パスになります。

5.1.3 破損したパスの参照を修正する

マッピング内でファイルレファレンスを追加または変更する場合にパスが解決されない場合、MapForce は、警告メッセージを表示します。こうすることにより、MapForce は、破損したパスへの参照を削減します。しかしながら、以下の場合破損したパスへの参照が起こる可能性があります:

- 相対パスを使用し、マッピングファイルをスキーマとインスタンスファイルを移動することなく、新しいディレクトリに移動する場合。
- マッピングファイルと同じディレクトリ内のファイル絶対パスを使用し、ディレクトリを他の場所に移動する場合。

このような状況が発生すると、MapForce は、このコンポーネントを赤でハイライトします。例:



破損したパスの参照

この場合の解決方法は、コンポーネントヘッダーをダブルクリックして、**「コンポーネント設定」**ダイアログボックス内で破損したパスを更新します (コンポーネント設定を変更するも参照)。

5.1.4 生成されたコード内のパスについて

コードをマッピングから生成する場合、または、MapForce Server 実行可能ファイル (mfx) に対してマッピングをコンパイルする場合 生成されたファイルは、MapForce で実行することはできず、選択されたターゲット環境で実行できるようになります。 (例 :RaptorXML Server、MapForce Server、または C# アプリケーション)。インスタンスファイルに対する相対ファイル名は、実行環境に対して相対となるよう解決されます。

従い、インスタンスまたはスキーマファイルに対してマッピングが相対パスを使用する場合、各ターゲット言語のベースパスに関しては以下のテーブルを考慮してください:

ターゲット言語	ベースパス
XSLT/XSLT2	XSLT ファイルのパス
XQuery*	XQuery ファイルのパス
C++、C#、Java*	生成されたアプリケーションの作業ディレクトリ

ターゲット言語	ベースパス
BUILT-IN* (MapForce でマッピングをプレビューする場合)	マッピング (.mfd) ファイルのパス
BUILT-IN* (MapForce Server でマッピングを実行する場合)	現在の作業ディレクトリ
BUILT-IN* (FlowForce Server の管理下で、MapForce Server でマッピングを実行する場合)	ジョブの作業ディレクトリまたは FlowForce Server の作業ディレクトリ

** MapForce Professional とEnterprise エディションで使用可能な言語*

必要な場合、マッピングのためにコードを生成する際、MapForce に全てのパスを相対から絶対パスに変換するように命令することができます。このオプションは、マッピングコードを、同じオペレーティングシステム内 (または MapForce Server 実行ファイル)で、または、マッピングにより使用される絶対パスへの参照がある他のオペレーティングシステムで、実行する際に役に立ちます。

生成されたコード内で全てのパスを絶対パスに変換するには、マッピング設定ダイアログボックスの**生成されたコード内でパスを絶対パスにする** チェックボックスを選択します (マッピング設定の変更を参照)。

コードを生成する際、チェックボックスが選択されていると、MapForce は、マッピングファイル (.mfd) のディレクトリをベースにした相対パスを解決し、生成されたコード内でこれらを絶対パスにします。この設定は、以下のファイルのパスに影響を及ぼします :

- すべてのファイルベースの コンポーネント型のための入力と出力インスタンスファイル
- マッピングコンポーネントとして、使用される SQLite データベースファイルへアクセス

チェックボックスが選択されていない場合、ファイルパスは、コンポーネント設定内で定義されたとおりの生成されたコードで書き込まれます。

5.1.5 コピーと貼り付けと相対パス

マッピングからコンポーネントをコピーし、貼り付ける場合は、スキーマファイルの相対パスを目的のマッピングのフォルダーに対して解決することができます。もし、パスが解決されない場合、相対パスを絶対パスにソースマッピングのフォルダーを使用して変更するようにプロンプトされます。最初に目的のマッピングに保存することが奨励されます。それ以外の場合、相対パスは、個人のアプリケーションフォルダーに対して解決されます。

5.2 マッピング接続の種類

マッピング接続を作成すると、オプションで接続（子アイテムを持つソースとターゲットアイテム）の種類を以下から選択することができます。

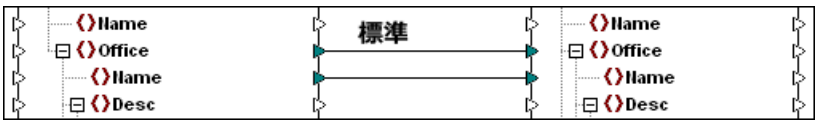
- ターゲット優先（標準）
- ソース優先（混合コンテンツ）
- 全てコピー（子アイテムのコピー）

接続の種類はマッピングにより生成された出力内の子アイテムのシーケンスを決定します。このセクションは各接続の種類と役に立つシナリオについての情報を提供します。

5.2.1 ターゲット優先マッピング

ターゲット優先（標準マッピング）は、MapForce におけるデフォルトのマッピング方法で、ターゲットスキーマ内のノードのシーケンスにより出力が決定されます。このマッピングの種類は大部分のシナリオに適しており、MapForce のデフォルトのマッピングとして設定されています。

標準マッピングは実線で表示されます。



ターゲット優先マッピングは（文字データや子要素などの）複合コンテキストを含むXML ノードをマップする場合適していない場合があります。例：

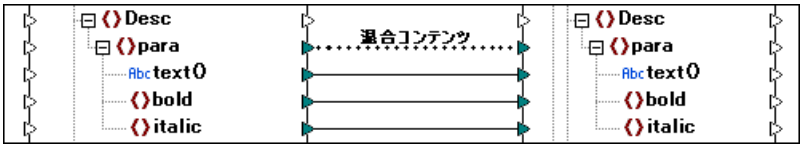
```
<p>This is our <i>best-selling</i> product.</p>
```

複合コンテンツが含まれるマッピングの場合、ソースファイル内で現れるアイテムのシーケンスを保持ため、ソース優先マッピングを使用することが奨励されます（*ソース優先マッピング* を参照）。

5.2.2 ソース優先マッピング

ソース優先（混合コンテンツ）マッピングを使うことで、テキストならびに子ノードを、XML **ソース**ファイルにある順序通りにマッピングすることができます。

- 混合コンテンツテキストノードコンテンツがサポートされます。
- 子ノードの順序はソースXML インスタンスファイルに依存します。



混合コンテンツのマッピングは点線により表示されます。

勿論、ソース優先（混合コンテンツ）マッピングをXML スキーマの**複合型**に適用することもできます。XML ソースファイルの順序に従って子ノードのマッピングが行われます。

ソース優先 /混合コンテンツマッピングでは以下がサポートされます：

マッピング

- **ソース**コンポーネントとして：
  - XML スキーマ複合型（mixed=true となっている混合コンテンツを含む）
  - データベースフィールドへ組み込まれたスキーマ内にある、XML スキーマ複合型（混合コンテンツを含む）
- **ターゲット**コンポーネントとして：
  - XML スキーマ複合型（混合コンテンツを含む）
  - データベースフィールドへ組み込まれたスキーマ内にある、XML スキーマ複合型（混合コンテンツを含む）

メモ：CDATA セクションはテキストとして扱われます。

5.2.2.1 混合コンテンツのマッピング

以下のサンプルでは**...\MapForceExamples\Tutorial** フォルダー以下にある**Tut-OrgChart.mfd**、**Tut-OrgChart.mfd.xml**、**Tut-OrgChart.mfd.xsd, Tut-Person.xsd** を使用します。

ソース XML インスタンス
このセクションで使用されている**Tut-OrgChart.XML** ファイルの一部を以下に示します。ここで注目したいのは "bold" ならびに "italic" 子ノードも含まれている "para" の混合コンテンツです。

para 要素には処理命令（<?sort alpha-ascending?>）だけではなく、コメントテキスト（<!--Company details... -->）も含まれていることに注目してください。これらのデータもマッピングすることができます。

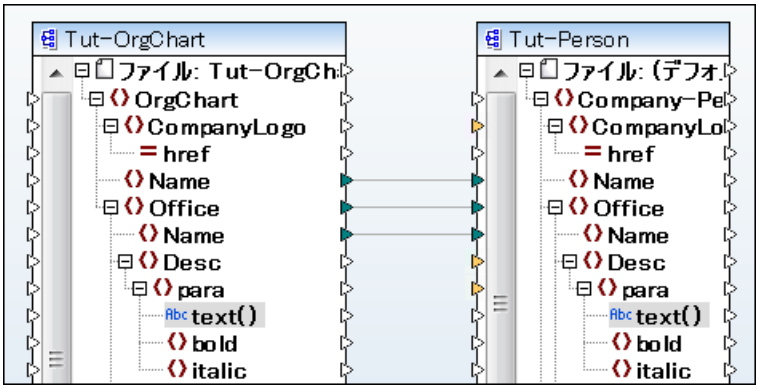
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp2 U (http://www.altova.com) by Mr. Nobody (Altova GmbH) -->
<OrgChart xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Tut-OrgChart.xsd">
  <CompanyLogo href="nanonull.gif"/>
  <Name>Organization Chart</Name>
  <Office>
    <Name>Nanonull, Inc.</Name>
    <Desc>
      <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull
develops nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999
saw the unveiling of the first prototype <bold>Nano-grid.</bold>The company hopes to expand
its operations <italic>offshore</italic>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>
```

XML インスタンスファイルにあるテキストの**シーケンス**ならびに bold/italic ノードは以下のような構成（順序）となっています：

```
<para> The company...
    <bold>Vereno</bold>in 1995 ...
    <italic>multi-core...</italic>February 1999

    <bold>Nano-grid.</bold>The company ...
    <italic>offshore...</italic>to drive...
</para>
```

初期のマッピング
Tut-OrgChart.mfd を開いたときのマッピングを以下に示します。



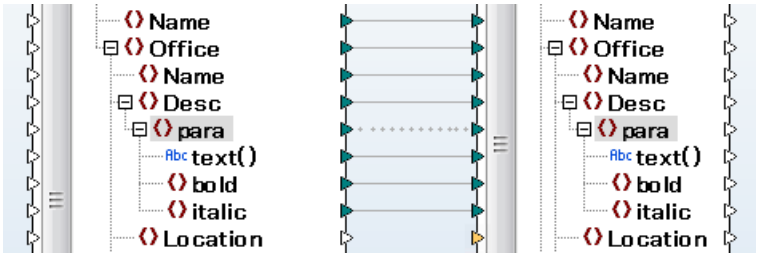
上のマッピングを出力

上に示されるマッピング結果を以下に示します。組織図 や個々のオフィス名が表示されます。

```
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
```

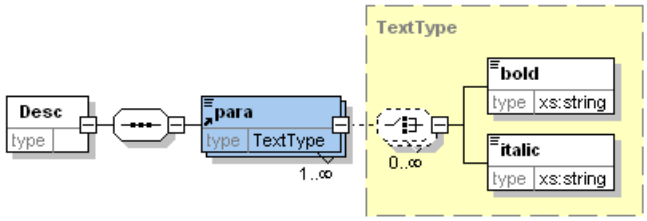
para 要素のマッピング

以下のスクリーンショットは混合コンテンツマッピングの例です。混合コンテンツの para 要素にコネクタが接続されており、**点線**によりそれが混合コンテンツであることを表しています。**text()** ノードにはテキストデータが含まれており、ターゲットコンポーネントへテキストを表示するには、このノードをマッピングする必要があります。



コネクタを右クリックしてプロパティを選択することで、コネクタに注釈を追加したり、ラベルを付与することができます。詳細については[接続の注釈](#) を参照ください。

以下のイメージは **Tut-OrgChart.xsd** スキーマファイルにある Desc 要素のコンテンツモデルを表しています。この例の場合、この定義はソースならびにターゲットスキーマの両方にて共通したものとなっています。



コンテンツモデルにある **para** 要素について、以下の事柄を理解できます：

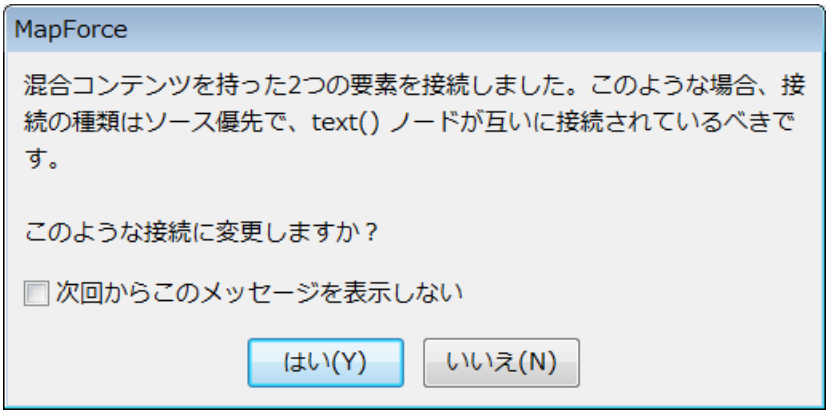
- **para** は "TextType" という型で、mixed="ture" の複合型です。

- **bold** ならびに**italic** 要素は両方とも "xs:string" 型となっており、この例では再帰的な型として定義されていません（つまり**bold** も**italic** も "textType" 型ではありません）。
- **bold** ならびに**italic** 要素は、**para** 内部においてどのような順序でも無制限の数だけ出現することができます。
- **para** 要素内では**bold** ならびに**italic** 要素とともに、任意のテキストを入力することができます。

アイテム間で混合コンテンツの接続を作成：

1. メニューオプションの「接続 | 子要素の自動接続」を選択して、有効にします（既に有効になっていない場合）。
2. ソーススキーマにて**para** アイテムから、ターゲットスキーマの**para** アイテムへの接続を行います。

   ソース優先のマッピングを行うかを尋ねるメッセージが表示されます。



3. 「はい」をクリックして混合コンテンツ接続を作成します。

メモ：
para は混合コンテンツであるため、この時点でメッセージが表示されます。自動接続オプションを無効にした状態で para アイテムを直接接続した際にも混合コンテンツのメッセージが表示されます。

para アイテムにある全ての子アイテムが接続されました。para アイテムへ接続しているコネクタが点線で表示され、それが混合コンテンツであることを表しています。

4. 出力タブをクリックして、マッピングの結果を確認します。

```
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<bold> Vereno</bold>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<italic>Europe</italic> was set up in Vienna. The team co
17     </Desc>
18   </Office>
19 </Company-Person>
20
```

5. 出力タブのアイコンバーにある**ワードラップ**アイコン をクリックして、全てのテキストを出力ウィンドウにて表示します。

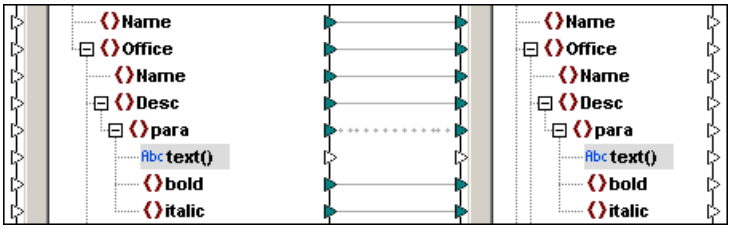
```
3     <Name>Organization Chart</Name>
4     <Office>
5       <Name>Nanonull, Inc.</Name>
6       <Desc>
7         <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull develops
      nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999 saw the
      unveiling of the first prototype <bold>Nano-grid.</bold>The company hopes to expand its
      operations <italic>offshore</italic>to drive down operational costs.
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<italic>Europe</italic> was set up in Vienna. The team
      consists of<bold> five research scientists </bold>and one administrative staff.</para>
17      </Desc>
18    </Office>
19  </Company-Person>
20
```

各オフィスに関する記述である混合コンテンツテキストが正しくマッピングされました。テキストだけでなくbold ならびにitalic タグコンテンツが、XML **ソース**ファイルにある通りにマッピングされました。

6. マッピングビューに切り替えます。

混合コンテンツからテキストノードを削除する：

1. **text()** ノードのコネクタをクリックして、Del キーを押下することで接続を削除します。



2. 出力タブをクリックしてマッピングの結果を確認します。

```
5       <Name>Nanonull, Inc.</Name>
6       <Desc>
7         <para>
8           <bold> Vereno</bold>
9           <italic>multi-core processors.</italic>
10          <bold>Nano-grid.</bold>
11          <italic>offshore</italic>
12        </para>
13        <para/>
14      </Desc>
15    </Office>
16    <Office>
17      <Name>Nanonull Europe, AG</Name>
18      <Desc>
19        <para>
20          <italic>Europe</italic>
21          <bold> five research scientists </bold>
22        </para>
23      </Desc>
```

結果：

- para 要素内にある全ての**テキスト**ノードが削除されました
- それまでマッピングされていたbold ならびにitalic 要素のテキストはまだ残っています。
- bold ならびにitalic アイテムのシーケンスもソースXML ファイルと同様のものとなっています。

処理情報とコメントのマッピング:

1. コンテンツ接続を右クリックして、**プロパティ**を選択します。
2. **ソースドライブ (混合コンテンツ)**から、**処理命令をマップ**と**コメントをマップ**のチェックボックスを選択します。

5.2.2.2 混合コンテンツのサンプル

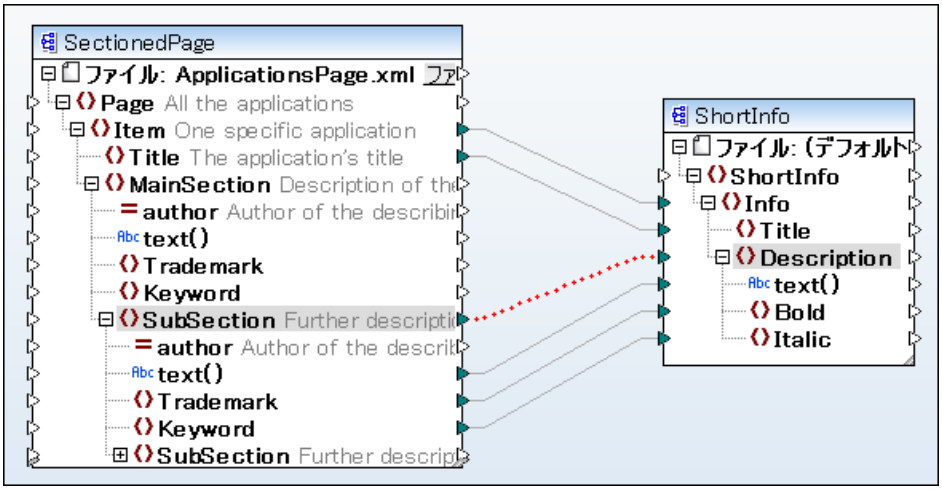
以下のサンプルでは **...\ MapForceExamples** フォルダー以下にある**ShortApplicationInfo.mfd** を使用します。

この例で使用するXML ソースファイルの一部を示します：

```
<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>
```

以下にマッピングを示します。以下の点に注意してください：

- "SubSection" アイテムのコネクタは混合コンテンツで、ターゲットXML/スキーマの詳細 アイテムへマッピングされています。
- text() ノード同士がマッピングされています。
- Trademark テキストがターゲットのBold アイテムへマッピングされています。
- Keyword テキストがターゲットのItalic アイテムへマッピングされています。



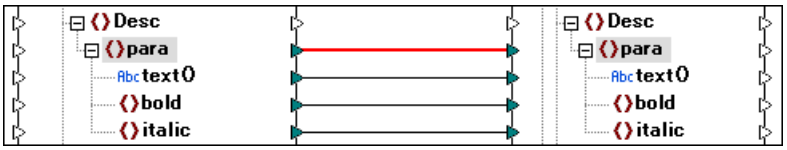
マッピング結果
各説明文の混合コンテンツテキストが正しくマッピングされ、テキストだけではなく、bold とitalic タグコンテンツがXML ソースファイルにある順序の通り、正しくマッピングされます。

```
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>
```

5.2.2.3　混合コンテンツアイテムに対して標準接続を使用する

上記のように、ソース優先の（標準ではない）接続は、通常、複合コンテンツノードからデータをマッピングする際に使用されます。それ以外の場合、結果する出力は、希望する結果ではない可能性があります。通常の標準（ターゲット優先）接続を使用して、複合コンテンツノードからデータをマッピングする場合の結果を確認するには、以下の手順を踏んでください：

1. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーからマッピング **Tut-OrgChart.mfd** を開きます。
2. ソース内の`para`ノードとターゲット内の`para`ノード間に接続を作成します。MapForce が接続をソース優先と定義するかを問うメッセージが表示されます。**【いいえ】**をクリックします（これにより、MapForce による提案を破棄し、通常の接続を作成します）。



メモ： 接続が上で表示されているように標準（ターゲット優先）であることを確認してください。すべてコピー接続が自動的に作成されると、接続を右クリックして、コンテキストメニューから **「ターゲット優先」（標準）**を選択します。

3. **「出力」**タブをクリックして、マッピングの結果を表示します。

```
<Office>
  <Name>Nanonull, Inc.</Name>
  <Desc>
    <para>The company was established inin 1995. Nanonull develops nanoelectronic techn
unveiling of the first prototype The company hopes to expand its operations to drive down opera
      <bold> Vereno</bold>
      <bold>Nano-grid.</bold>
      <italic>multi-core processors.</italic>
      <italic>offshore</italic>
    </para>
    <para>White papers and further information will be made available in the near future.
    </para>
  </Desc>
</Office>
<Office>
```

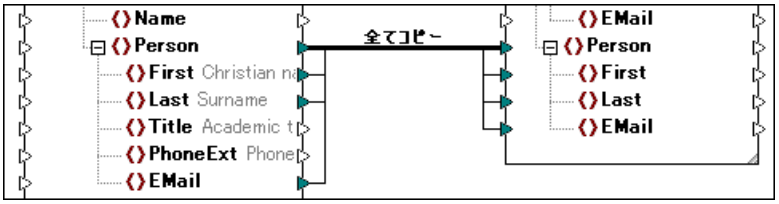
上に示されるように、混合コンテンツのアイテムに対して標準マッピングを適用すると、以下のような結果となります：

- `text()`ソースアイテムのコンテンツがターゲットにコピーされます。しかしながら、出力内の子ノード（この場合、`bold`と`italic`）のシーケンスは、ターゲット XML スキーマ内のシーケンスに対応します。すなわち、子ノード（この場合、`bold`と`italic`）は、複合コンテンツノードテキストの後に表示されます。
- それぞれの`para`要素に対して、MapForce は、`text()`ノードを最初にマップし、すべての`bold`アイテムと、最後に全ての`italic`アイテムをマップします。この結果、複数の`bold`と`italic`アイテムは、互いに積み上げられて表示されます。各アイテムのコンテンツは、接続がソースから存在する場合に表示されます。

5.2.3 全てコピー接続

全てコピーという種類の接続方法では、作業領域にあるマッピングをより単純なものとして、ソースならびにターゲットコンポーネントにある**全ての**同一アイテムを自動的に接続します。つまり、ソースならびにターゲットコンポーネントにおける型に従うかたちで、ソースならびにターゲットアイテムの**型**が**同一**の場合か、またはターゲットの型が xs:anyType の場合、該当する全てのソースアイテムがターゲットコンポーネントへ**コピー**されます。

*ソースならびにターゲットの**型**が**同一で無い**場合、かつターゲットの型がxs:anyType で無い場合、ソースデータは同一の階層にある同一名のアイテムにマッピングされます。ターゲットアイテムの名前が異なる場合、そのターゲットアイテムに対するマッピングは作成されません。*



全てコピー接続

マッピング上の2つの構造間にマッピングを描く場合シースとターゲット構造に互換性があると検出した場合、MapForce は、「全てコピー」接続を自動的に作成します（これは、両方の構造が同じ型、または、ターゲットがソース型のサブ型であることを意味します）。マッピングのランタイムでは、子を含む全てのインスタンスデータはソースからターゲットに再帰的にコピーされます。

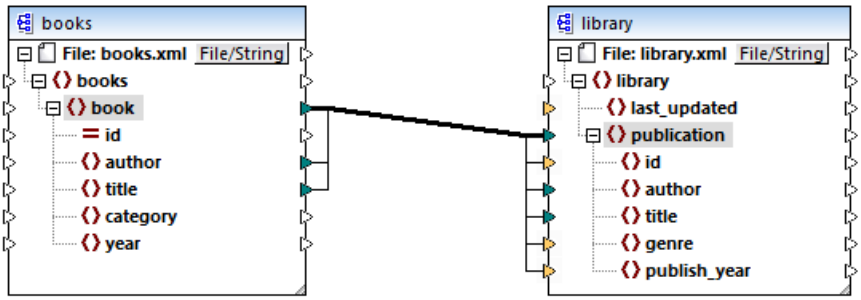
「全てコピー」接続を手動で作成するには、子アイテムを持つ同類の2つのノード間の既存の接続を右クリックし、コンテキストメニューから**全てコピー（子アイテムのコピー）**を選択します。

次に注意してください：

- 「全てコピー」接続が意味を成さない、または、サポートされていないコンテキストでは、この種類の接続を手動で作成することはできません。
- 「全てコピー」接続をXML/スキーマ コンポーネントの`root` 要素に対して作成することはできません。
- ユーザー定義関数のスキーマとパラメーター間に「全てコピー」接続を作成する場合、2つのコンポーネントは同じスキーマをベースにしている必要があります。しかしながら、同じルート要素を持つ必要はありません。

「全てコピー」接続の例は次のステップで作成することができます：

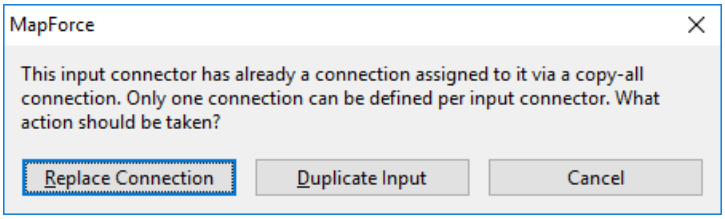
1. 新規マッピングを作成します。
2. **「挿入」**メニューから**XML スキーマ/ファイル**をクリックし、フォルダー**<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial** 内にある**books.xml** ファイルを参照してください。
3. **「挿入」**メニューから**XML スキーマ/ファイル**をクリックし、フォルダー**<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial** 内にある**library.xsd** ファイルを参照してください。
4. 「books」コンポーネントの`book` ノードと「library」コンポーネントの`publication` ノード間にマッピング接続を描きます。
5. 新規の接続を右クリックして、コンテキストメニューから**全てコピー（子アイテムのコピー）**を選択します。



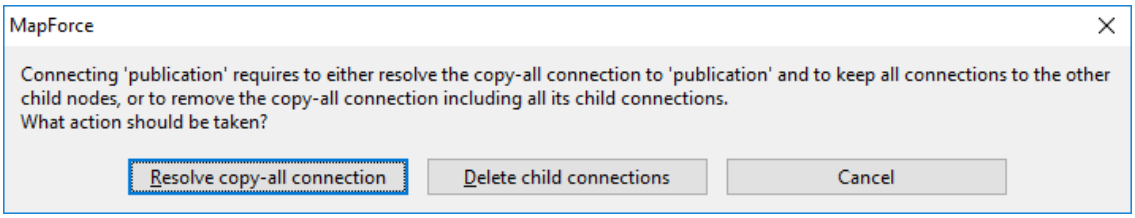
ソースとターゲット構造間に差異が存在する場合、「全てコピー」接続が列挙します。マッピングのランタイムでは、(要素と属性などの)ソースアイテムは、ターゲット型内に存在するものだけがコピーされます。これは再帰的に繰り返されます。

例えば、上のマッピングでは、2つの構造(`author` と`title`)の2つの子アイテムが同一です、ですから、ターゲットにマップすることができます。アイテム`id`は、ソース属性内の属性とターゲット内の要素であるため、自動的には含まれません。マップする必要がある場合、例えば、`category` から`genre`, へマップする場合、異なるアイテムのため、「全てコピー」接続を使用することはできません。

入力コネクタが(コンポーネントの横の小さい三角で表示されている)「全てコピー」接続を受け取る場合、他の接続を受け入れることはできません。上のサンプルでは、`category` と`genre` 間の接続を作成する場合、MapForce 置換え、または、入力の複製を行うようにプロンプトします。



ここでは必要ありませんが、入力の複製は、ターゲットが1つ以上の入力からデータを受け入れる場合に役に立ちます(次を参照：入力の複製)。「全てコピー」接続を置き換える場合、メッセージボックスは、「全てコピー」接続を解決、または、削除するようにプロンプトします。

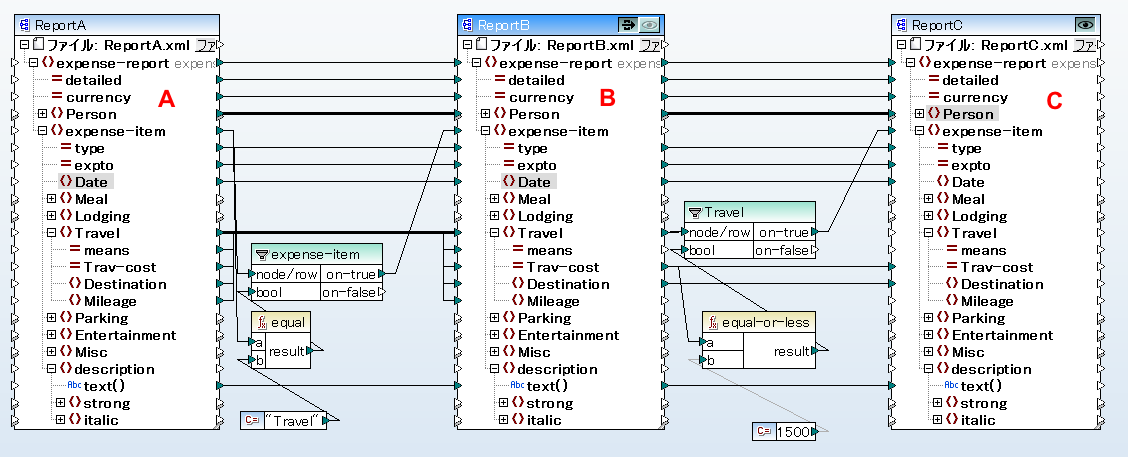


「全てコピー」接続を対応する子アイテムの標準のターゲット優先 接続と置換えるには、**全てコピー接続を解決する**をクリックします。「全てコピー」接続を削除するには、**「子接続の削除」**をクリックします。

5.3 チェーンマッピング

MapForce ではチェーン構造により複数のコンポーネントを含むようなマッピングがサポートされます。チェーン構造のマッピングとは、少なくとも 1 つのコンポーネントがソースかつターゲットコンポーネントとして機能するマッピングのことです。このようなコンポーネントの出力は、それ以降のマッピングで入力として使用されます。このようなマッピングのことを「中間」コンポーネントと呼びます。

例えば、下に示されるマッピングは、(XML 書式の)2つの段階で処理される経費報告書を表示しています。A からB へのマッピングの部分は、"Travel" とマークされる経費のみをフィルターします。B からC へのマッピングの部分は、1500以下の "Travel" 経費のみをフィルターします。コンポーネントB は、入力と出力接続の両方が存在するため、"中間 "コンポーネントです。このマッピングは、次のパスで見つけることができます :**<Documents>\Altova\MapForce2017 \MapForceExamples\Tutorial\ChainedReports.mfd**。



ChainedReports.mfd

"パススルー"と呼ばれる機能がチェーンマッピングに搭載されています。"パススルー"は、出力ウィンドウ内でチェーンマッピングの各段階による出力を確認することのできるプレビュー機能です。例えば、上のマッピングでは、A からB への、および、B からC への、結果するXML 出力をプレビューして、保存することができます。

メモ: XML、CSV、TXT ファイルといったファイルをベースにした「中間」コンポーネントだけが「パススルー」機能を搭載します。データベースコンポーネントを中間コンポーネントとして使用することはできますが、パススルーコンポーネントは表示されません。プレビューやコードの生成を行うたびに、中間コンポーネントは再度生成されますが、生成を行う前に一度削除しなければならないため、データベースにてこのような操作を行うことはできません。

MapForce Server または、生成されたコードによりマッピングが実行される場合、マッピングのチェーン全体が実行されます。チェイン内の各ステップで必要な出力ファイルをマッピングは生成し、マッピングのチェーンのステップの出力が次のマッピングステップへ入力として転送されます。

中間コンポーネントのために、動的なファイル名を生成することもできます。コンポーネントが対応して構成されていることを条件に、マッピングから "File:" アイテムへの接続を受け入れることができます。更に詳しい情報に関しては、次を参照してください:複数の入力、および、出力ファイルを動的に処理する。

プレビューボタン

コンポーネントB ならびにC にはプレビューボタンが表示されます。この機能を使用することによりMapForce 内でB の中間マッピング結果のプレビューだけではなく、チェーン構造の最終結果をプレビューすることができます。対応するコンポーネントのプレビューボタンをクリックし、出力ボタンをクリックすることでマッピング結果を確認することができます。

パススルーボタンが有効化されている中間 "コンポーネントをプレビューすることはできません。同時にデータをプレビューし、パススルーすることは意味を成さないため、プレビューボタンは、自動的に無効化されます。このようなコンポーネントの出力を確認するためには、パススルーボタンをクリックして、無効化してから、プレビューボタンをクリックします。

パススルーボタン
中間コンポーネントB には、更に「パススルー」という名前のボタンがコンポーネントのタイトルバーに表示されます。

パススルーボタンが有効 になっている場合、コンポーネントA からコンポーネントB へ、そしてコンポーネントC へというかたちで全てのデータをプレビューウィンドウへマッピングします。2つの結果が生成されます。

- マッピングコンポーネントA の結果が中間コンポーネントB へマッピングされます。
- 中間コンポーネントB の結果がターゲットコンポーネントC へマッピングされます。

パススルーボタンが**無効** になっている場合、マッピングチェーンの一部だけが実行されます。生成されるマッピング結果は、(コンポーネントB とC のうち) どちらのプレビューボタンがアクティブになっているかに依存します。

- コンポーネントB のプレビューボタンがアクティブになっている場合、コンポーネントA からB へのマッピング結果が生成されます。マッピングチェーンはコンポーネントB で停止し、プレビューを行うのにコンポーネントC は全く使用されません。
- コンポーネントC のプレビューボタンがアクティブになっている場合、中間コンポーネントB からC へのマッピング結果が生成されます。パススルー機能が無効になっているため、コンポーネントB に対する自動チェーンが中断されます。マッピング右側にあるチェーンだけが実行され、コンポーネントA は使用されません。

"パススルー"ボタンが無効化されている場合、中間コンポーネントは、"入力 XML ファイル"と"出力 XML ファイル"フィールド内で同一のファイル名を持つ必要があります。これにより、B とC の間のマッピングの部分をプレビューする際に、出力として生成されるファイルがA とB の間のマッピングを入力と使用することができます。また、生成されたコード内で、または、MapForce Server 実行内で、マッピングのチェーンが壊れないことを保証することができます。

前記のように、マッピング MapForce Server、または、生成されたコードにより実行された場合、すべてのコンポーネントの出力が生成されます。この場合、コンポーネントB のパススルーボタン、および現在選択されているプレビューコンポーネントは破棄されます。上のマッピングを例にして、2つの結果ファイルは以下のように生成されます:

1. A からB へのマッピングコンポーネントから結果する出力ファイル
2. B からC へのマッピングコンポーネントから結果する出力ファイル

次のセクションであるサンプル:パススルーが有効な場合 とサンプル:パススルーが無効な場合 は、どのようにソースデータがパススルーボタンが有効化、または、無効化されている場合に、転送されるかを説明しています。

5.3.1 サンプル:パススルーが有効な場合

このサンプルで使用されているマッピング (**ChainedReports.mfd**) は、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にあります。このマッピングは、旅費の経費を含む **ReportA.xml** という名前のXML ファイルを処理します。簡素化のために、名前空間宣言と`expense-item` 要素の一部は省略されています:

```
<?xml version="1.0" encoding="UTF-8"?>
<expense-report currency="USD" detailed="true">
   <Person>
      <First>Fred</First>
      <Last>Landis</Last>
      <Title>Project Manager</Title>
```

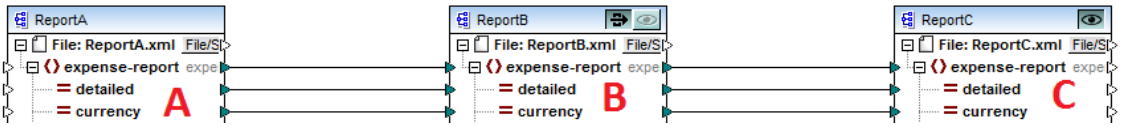
```
      <Phone>123-456-78</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item type="Travel" expto="Development">
      <Date>2003-01-02</Date>
      <Travel Trav-cost="337.88">
         <Destination/>
      </Travel>
      <description>Biz jet</description>
   </expense-item>
   <expense-item type="Lodging" expto="Sales">
      <Date>2003-01-01</Date>
      <Lodging Lodge-cost="121.2">
         <Location/>
      </Lodging>
      <description>Motel mania</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>
```

ReportA.xml

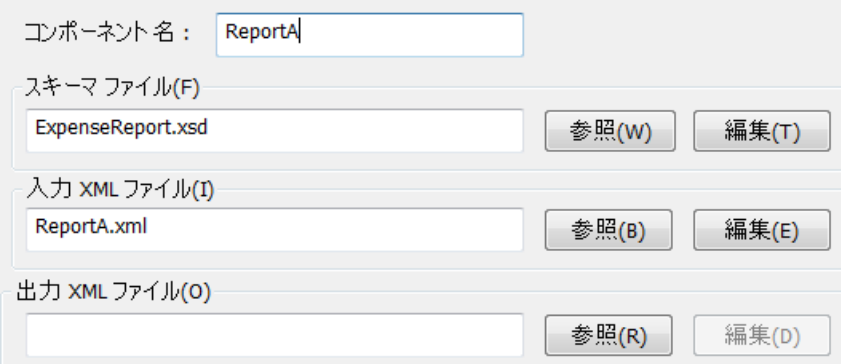
マッピングの目的は、上のファイルを基にして、2つの更なるレポートを生成することです：

- **ReportB.xml** - このレポートは、"Travel" 型の旅費のみを含みます。
- **ReportC.xml** - このレポートは、1500 を超えない "Travel" 型の旅費のみを含みます。

この目的を達成するためには、マッピング (コンポーネントB) の中間コンポーネントでは、表示されているとおりパススルーボタン が有効化されています。これによりマッピングは段階的に実行されます：A からB は、そして、B からC へ実行されます。中間コンポーネントにより作成される出力は、B とC の間のマッピングのために使用されます。



マッピングのチェーン内の各段階で生成される出力ファイルは、各コンポーネントの設定により決定されます。 (コンポーネント設定を開くには、右クリックし、コンテキストメニューから**プロパティ**を選択します)。具体的には、最初のコンポーネントは、**ReportA.xml** という名前のXML ファイルからデータを読み取るように構成されています。ソースコンポーネントであるからであり、**出力 XML ファイル** フィールドは関連性がないため、空白のままです。



ソースコンポーネントの設定

下に示されるように、2番目のコンポーネント (**ReportB**) は、**ReportB.xml** という名前の出力ファイルを作成するように構成されています。**入力 XML ファイル** フィールドは、灰色で表示されています。(このサンプル内で示されているように) パススルーが有効化されている場合、中間コンポーネントの **入力 XML ファイル** フィールドは、自動的に無効化されます。マッピングを実行するために入力ファイル名が存在する必要はなく、マッピング内のこの段階で生成される出力は、一時的なファイル内に保管され、マッピングの後の段階で再利用することができます。また、(下で示されるように) **出力 XML ファイル** が定義されている場合、中間出力ファイルのファイル名のために使用されます。**出力 XML ファイル** が定義されていない場合、デフォルトのファイル名が自動的に使用されます。



中間コンポーネントの設定

最後に、3番目のコンポーネントは、**ReportC.xml** という名前の出力ファイルを生成するように構成されています。これはターゲットコンポーネントであるため、**入力 XML ファイルフィールド** には関連性はありません。

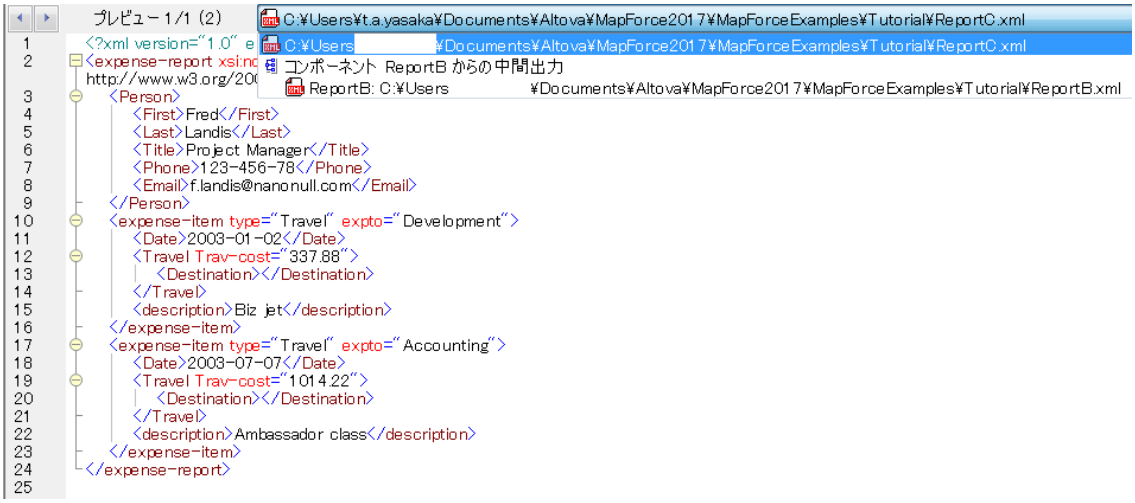


ターゲット コンポーネントの設定

マッピングウィンドウ内の**出力**タブをクリックして、マッピングをプレビューする場合は、2つのファイルが出力に表示されます：

1. **ReportB.xml** は、A からB へのマッピングの結果を表します。
2. **ReportC.xml** は、B からC へのマッピングの結果を表します。

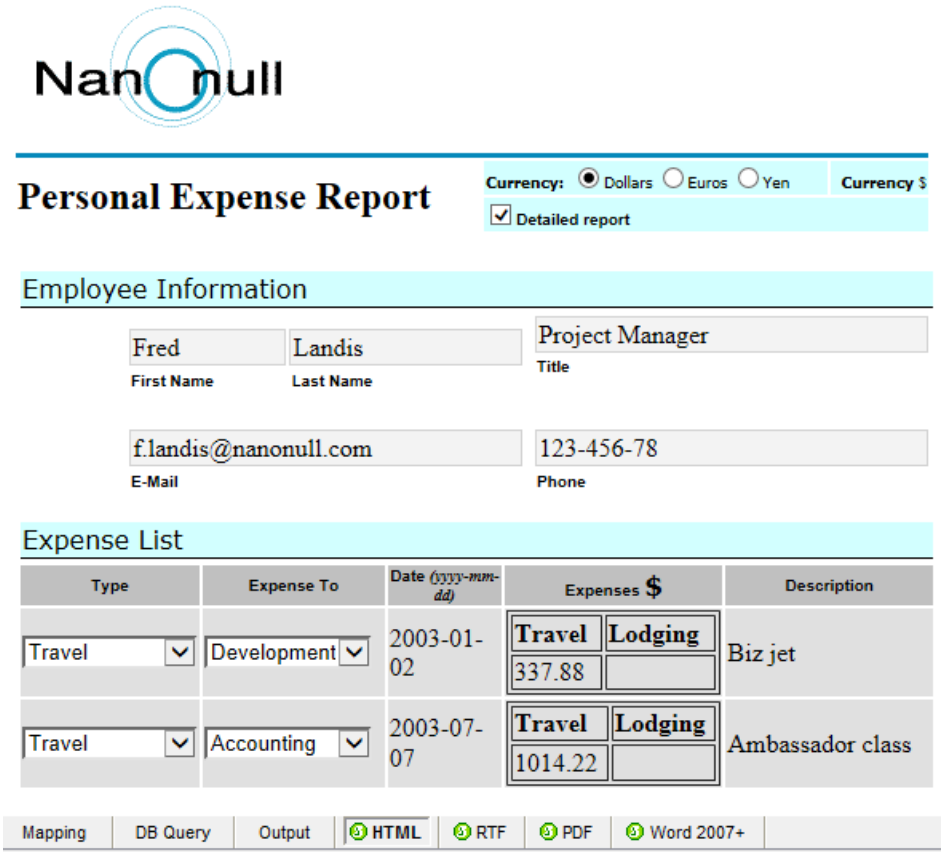
2件の生成された出力ファイルから選択するには、出力ウィンドウ内に表示されているファイルを矢印ボタンをクリックして、または、ドロップダウンリストからエントリを選択します。



生成された出力ファイル

MapForce によりマッピングが実行される場合、(**「ツール| オプション| 一般」**から構成することのできる) 設定「出力ファイルに直接書き込む」により、中間ファイルが一時的なファイル、または、物理的ファイルとして保存されるかを決定します。MapForce 内でマッピングが直接プレビューされる場合のみ、これは妥当であることに注意してください。このマッピングが MapForce Server により、または、生成されたコードにより実行された場合、実際のファイルは、マッピングのチェーンの各段階で生成されます。

StyleVision がインストールされている場合、そして、StyleVision Power Stylesheet (SPS) ファイルがターゲットコンポーネントに (このサンプル内で示されているように) 割り当てられている場合、最後のマッピング出力をHTML、RTF、PDF、またはWord ファイルとして確認 (保存) することができます。MapForce 内でこの出力を表示するには、対応する名前を持つタブをクリックしてください。

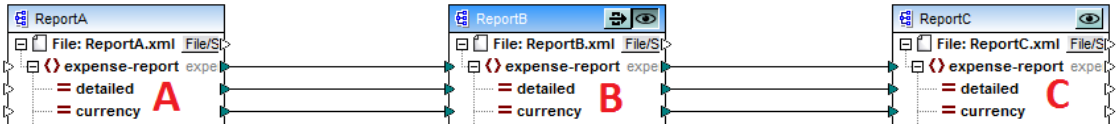


生成されたHTML 出力

マッピングのチェーンの最後のターゲットコンポーネントの出力のみが表示されることに注意してください。中間コンポーネントのStyleVision 出力を表示するには、パススルー ボタンを無効化して中間コンポーネントを (サンプル：パススルーが無効な場合 内で表示されているとおり)プレビューする必要があります。

5.3.2 サンプル :パススルーが無効な場合

このサンプル (**ChainedReports.mfd**) で使用されているマッピングは、**<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial** フォルダー内で見つけることができます。このサンプルは、中間コンポーネントのパススルーボタン が無効化されている場合、出力がどのように生成されるかを説明しています。



サンプル：パススルーが有効な場合 で説明されているように、マッピングの目的は、2件の個別のレポートを生成することです。前のサンプルで、パススルー ボタン は有効化されており、両方のレポートは期待されるとおり生成され、**出力** タブ内で確認することができました。しかしながら、レポートの1件 (**ReportB.xml** または**ReportC.xml**)のみをプレビューする場合、パススルー ボタン ( ) は無効化されている必要があります。更に具体的には、パススルー ボタンを無効化することは、次の目的を達成するためには有効な場合があります：

- B からC へのマッピングを無視して、A からB により生成された部分をプレビューする。

- A からB へのマッピングを無視して、B からCにより生成された部分をプレビューする。

上に示されるようにパススルーボタンを無効化すると、**ReportB** または**ReportC** を選択することができます (両方に ボタンが表示されます)。

パススルーボタンの無効化により、中間コンポーネントがどの入力ファイルを読み取るかを選択することができます。多くの場合、これは**出力 XML ファイル** フィールド (このサンプル内で示されているように)内で定義されたファイルと同じです。



中間コンポーネントの設定

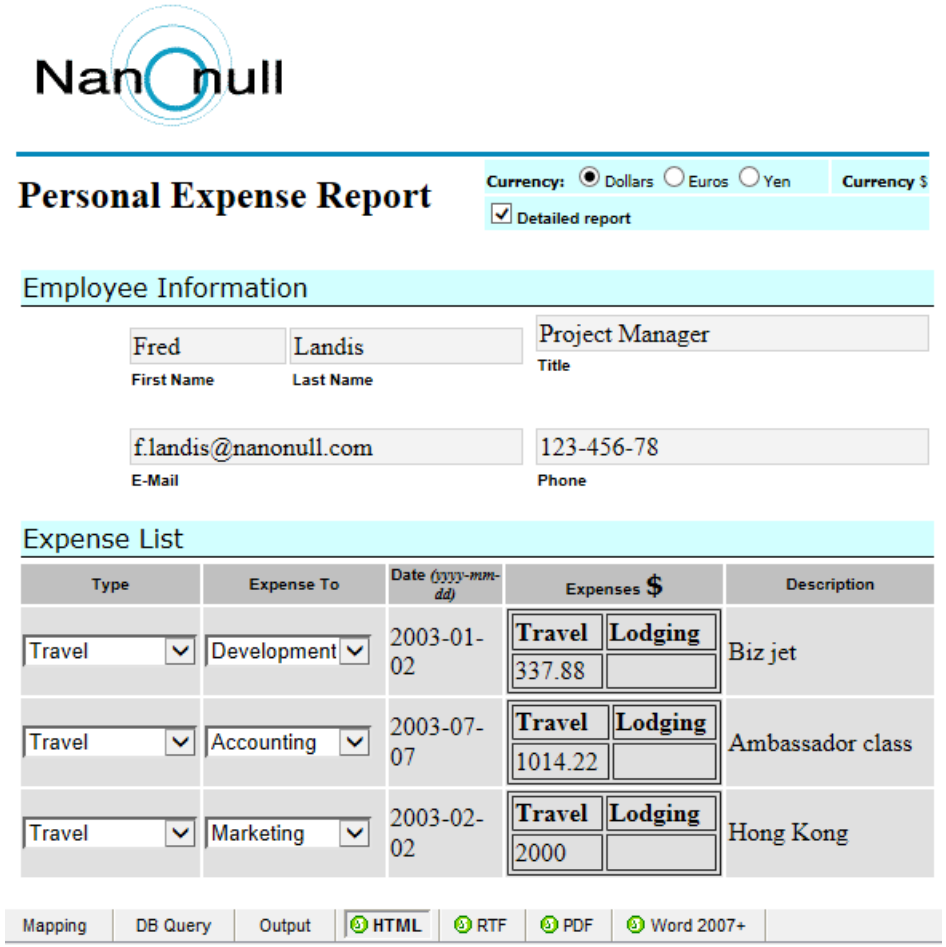
中間コンポーネント上に同じ入力と出力 ファイルが存在することは、マッピングからコードを生成する場合、または、マッピングをMapForce Server 上で実行する場合、特に重要です。前記のように、この環境は、マッピングチェーン内の各コンポーネントにより作成される出力により生成されます。ですから、中間コンポーネントが処理するファイル (この場合は **ReportB.xml**) を受け取り、異なるファイルを検索するよりも、同じファイルを次のマッピングに送ることは意味があります。(パススルーボタンが無効化されており)中間コンポーネント上で同じ入力と出力ファイル名が存在しないと、生成されたコード内で、または、MapForce Server 実行中に「システムが指定されたファイルを見つけることができません」などのエラーが発生する場合があります。

3番目のコンポーネント (**ReportC**) のプレビューボタン をクリックし、MapForce 内でマッピングをプレビューしようと試みると、エラーが発生します。上記の設定に従い、**ReportB.xml** という名前のファイルが入力として期待されるため、これは予想されている振る舞いです。 しかしながら、マッピングは、(パススルーボタンが有効化されておらず、B からC へのマッピングの部分のみが実行されているため)まだこの名前のファイルは生成されていません。この問題は簡単に解決することができます：

1. 中間コンポーネントのプレビューボタンをクリックします。
2. マッピングをプレビューするために、**出力** タブをクリックします。
3. マッピング (<**Documents**>**\Altova\MapForce2017\MapForceExamples\Tutorial**) と同じフォルダー内に結果出力ファイルを**ReportB.xml** として保存します。

3番目のコンポーネント (**ReportC**) のプレビューボタンをクリックすると、エラーは表示されなくなります。

パススルーボタンが無効化されている場合、関連したStyleVision Power StyleSheet (SPS) ファイルを持つ各コンポーネントのためのStyleVision-により生成された出力をプレビューすることができます。特に、(最後のレポートに加え)中間 レポートのHTML バージョンを確認することもできます ：



NanoNull

Personal Expense Report

Currency: (•) Dollars ( ) Euros ( ) Yen Currency $

[x] Detailed report

Employee Information

First Name	Last Name	Title
Fred	Landis	Project Manager

E-Mail	Phone
f.landis@nanonull.com	123-456-78

Expense List

Type	Expense To	Date (yyyy-mm-dd)	Expenses $		Description
			Travel	Lodging	
Travel	Development	2003-01-02	337.88		Biz jet
Travel	Accounting	2003-07-07	1014.22		Ambassador class
Travel	Marketing	2003-02-02	2000		Hong Kong

Mapping | DB Query | Output | HTML | RTF | PDF | Word 2007+

中間コンポーネントのHTML 出力

5.4 複数の入力または出力ファイルを動的に処理

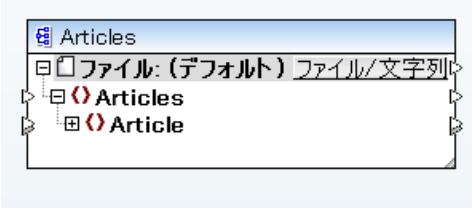
マッピングが実行される際、MapForce に複数のファイルを処理するように構成することができます (例 : ディレクトリ内の全てのファイル)。この機能を使用して、以下のようなタスクを解決することができます :

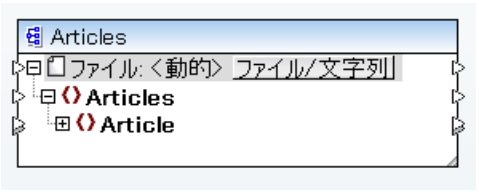
- マッピングに処理する入力ファイルのリストを与えます。
- 単一の出力ファイルの代わりにファイルのリストのマッピング出力を生成します。
- 入力と出力ファイル名がランタイムで定義されている箇所のマッピングアプリケーションを生成します。
- ファイルのセットを他のフォーマットに変換します。
- 大きなファイル (またはデータベース) を小さなパーツに分割します。
- 複数のファイルを1つの大きなファイルにマージする (または、データベースにロードする)

MapForce コンポーネントに複数のファイルを処理するように以下の方法で構成することができます :

- コンポーネント設定で必要とされる入力または出力ファイルに固定されたファイル名の代わりにワイルドカード文字を使用して、パスを与えます。(コンポーネント設定を変更するを参照)。 すなわち、ワールドカード * と ? をコンポーネント設定ダイアログボックスに入力し、MapForce は、マッピング実行される際に対応するパスを解決します。
- パスを動的に提供する、コンポーネント シーケンスルートノードに接続します。 (例 : `replace-fileext` 関数の結果)。マッピングが実行されると、MapForce は、全ての入力ファイルを動的に読み込み、また全ての出力ファイルを動的に生成します。

目的により異なりますが、同じマッピングで1つまたは両方のアプローチを取ることができます。しかしながら、同じコンポーネント上で同時に両方のアプローチを使用することは、あまり意味をなしません。特定のコンポーネントのために、MapForce がどのアプローチを使用するかを命令するには、コンポーネントのルートノードの横の **「ファイル」** (File) または **「ファイル文字列」** (File/String) ボタンをクリックします。このボタンにより以下の振る舞いを指定することができます :

コンポーネント設定からファイル名を使用する	コンポーネントが1つまたは複数のインスタンスファイルを処理する場合、このオプションは MapForce にコンポーネント設定ダイアログボックスで定義されたファイルを処理するように命令します。 このオプションを選択すると、ルートノードには入力コネクタが与えられません。  入力または出力ファイル in コンポーネント設定ダイアログボックス内でまだ入力または出力ファイルを指定していない場合、ルートノードの名前は **ファイル : (デフォルト)** です。それ以外の場合、ルートノードは、出力ファイル名、セミコロン (;) が後についた入力ファイル名を表示します。 入力ファイル名が出力ファイルと同じ場合、ルートノードの名前として表示されます。

	 *このオプションまたは「マッピングから与えられた動的なファイル名を使用する」オプションを選択することができます。*
マッピングから与えられた動的なファイル名を使用する	このオプションは、コンポーネントのルートノードの値に接続することにより、マッピングエリアで定義するファイル名を処理するようにMapForce に命令します。 このオプションを選択すると、ルートノードがマッピングの実行中動的に処理されるファイル名を与える入力コネクタの値に接続します。ファイル名 in コンポーネント設定ダイアログボックス内に定義されたファイル名を持つ場合、これらの値は無視されます。 このオプションが選択された場合、ルートノードの名前は以下として表示されます **File: <dynamic>**。  *コンポーネント設定からファイル名を使用するオプション*と共にこのオプションは相互排他的です。
XML に文字列を解析、JSON に文字列を解析、CSV に文字列を解析、FLF に文字列を解析、EDI に文字列を解析	このオプションがオンになると、コンポーネントが文字列の値をルートノードへの入力として受け入れ、それぞれ、XML、JSON、CSV、FLF、またはEDI 構造に変換します。詳細に関しては、文字列の解析とシリアル化を参照してください。
XML を文字列にシリアル化する、JSON を文字列にシリアル化する、CSV を文字列にシリアル化する、FLF を文字列にシリアル化する、EDI を文字列にシリアル化する	このオプションがオンになると、コンポーネントは構造を入力として受け入れ、文字列に変換します。入力構造は、それぞれXML、JSON、CSV、固定長フィールド、またはEDI であることができます。詳細に関しては、文字列の解析とシリアル化を参照してください。

複数の入力または出力ファイルを以下のコンポーネントのために定義することができます：

- XML ファイル
- テキストファイル（CSV*, FLF* ファイルとFlexText** ファイル）
- EDI ドキュメント**
- Excel スプレッドシート**
- XBRL ドキュメント**

** MapForce Professional Edition 必須*
*** MapForce Enterprise Edition 必須*

以下のテーブルは、 MapForce 言語内での動的な入力と出力ファイルおよびワイルドカードへのサポートについて説明されています。

ターゲット言語	動的入力ファイル名	入力ファイル名のためにサポートされているワイルドカード	動的な出力ファイル名
XSLT 1.0	*	XSLT 1.0 によりサポートされていません	XSLT 1.0 によりサポートされていません
XSLT 2.0	*	*(1)	*
XQuery	*	*(1)	XQuery によりサポートされていません
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILT-IN	*	*	*

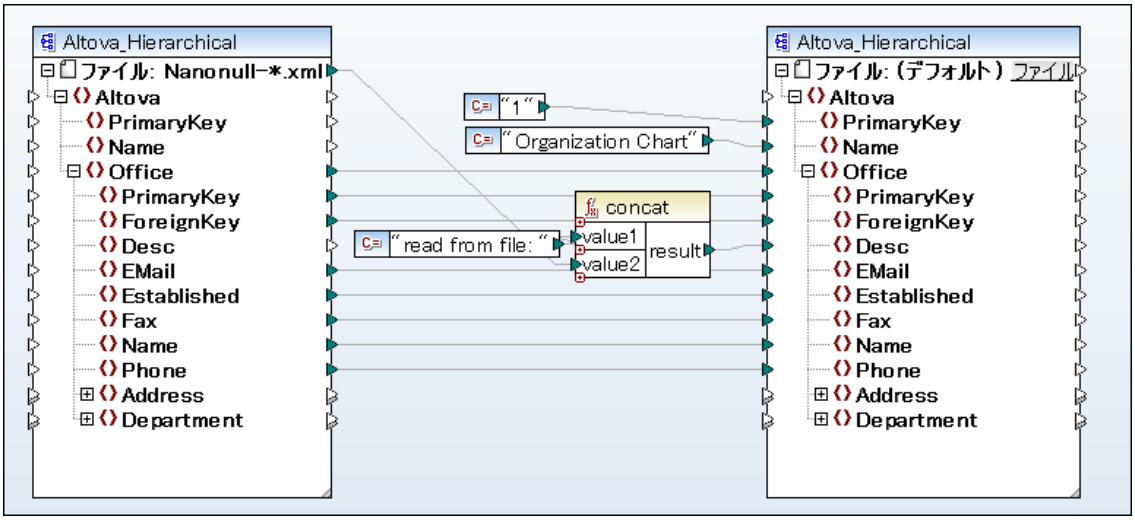
レジェンド:

*	サポートされている
(1)	`fn:collection` 関数を使用します。XSLT 2.0 とXQuery エンジンの実装は、ワイルドカードを解決します。他のエンジンは異なる場合があります。RaptorXML Server エンジンを使用してのXSLT 1.0/2.0 とXQuery コード変換方法に関しては、 XSLT 1.0 または 2.0 コードの生成 とXQuery 1.0 コードの生成 を参照してください。

5.4.1 複数の入力ファイルを単一の出力ファイルにマップする

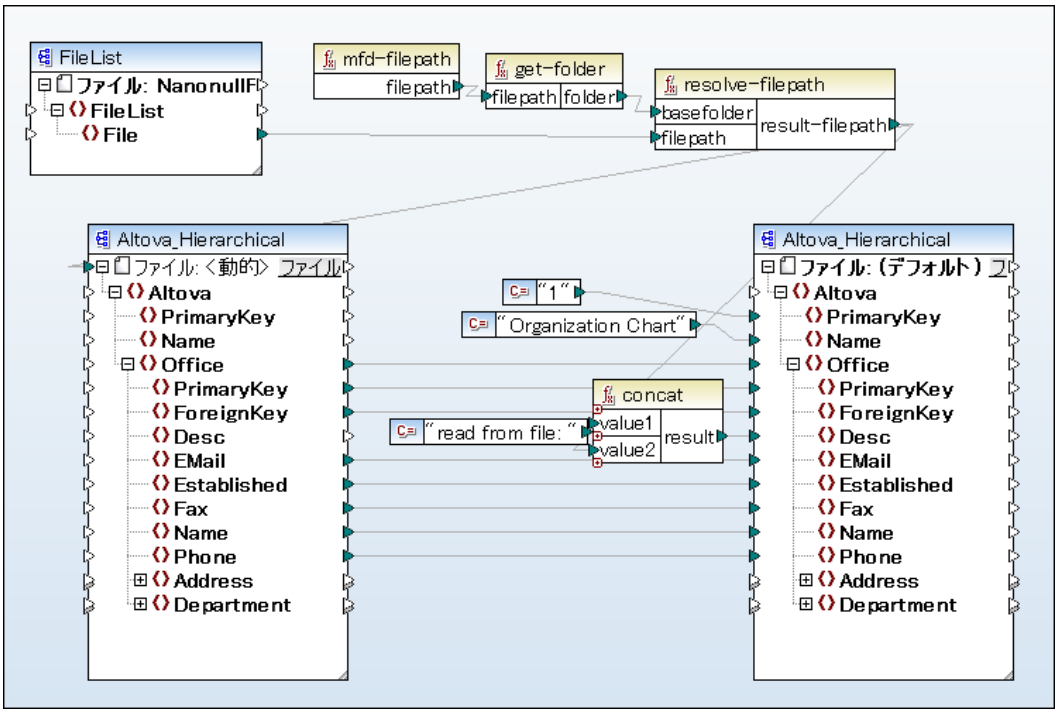
複数の入力ファイルを処理するには、以下を行います:

- コンポーネント設定ダイアログボックスに入力ファイルとしてワイルドカード (*または ?) 付きのファイルパスを入力します。一致する全てのファイルが処理されます。下のサンプルは、入力 XML ファイルフィールド内でマッピング入力として名前が "nanonull-" で開始する全てのファイルを提供する* ワイルドカード文字を使用します。ターゲットコンポーネントに動的コネクタが存在せず、ソースコンポーネントはワイルドカード* を使用して複数のファイルにアクセスしているため、複数の入力ファイルは**単一**の出力ファイルとしてマージされます。ターゲットコンポーネント内のルートノードの名前は **File: <default>** です。これはコンポーネント設定ダイアログボックス内で出力ファイルパスが定義されていないことを示します。複数のソースファイルは、ですから、ターゲットドキュメント内で追加されます。



MergeMultipleFiles.mfd (MapForce Basic Edition)

- *ソースコンポーネントのファイルノードに文字列のシーケンスをマップします。シーケンス内の各文字列が1つのファイル名を表します。文字列は自動的に解決されるワイルドカードを含むこともできます。XML ファイル、データベース テキストフィールド、テキストファイル (CSV または固定長) または Excel シートなどのコンポーネントがファイル名のシーケンスを提供します。*

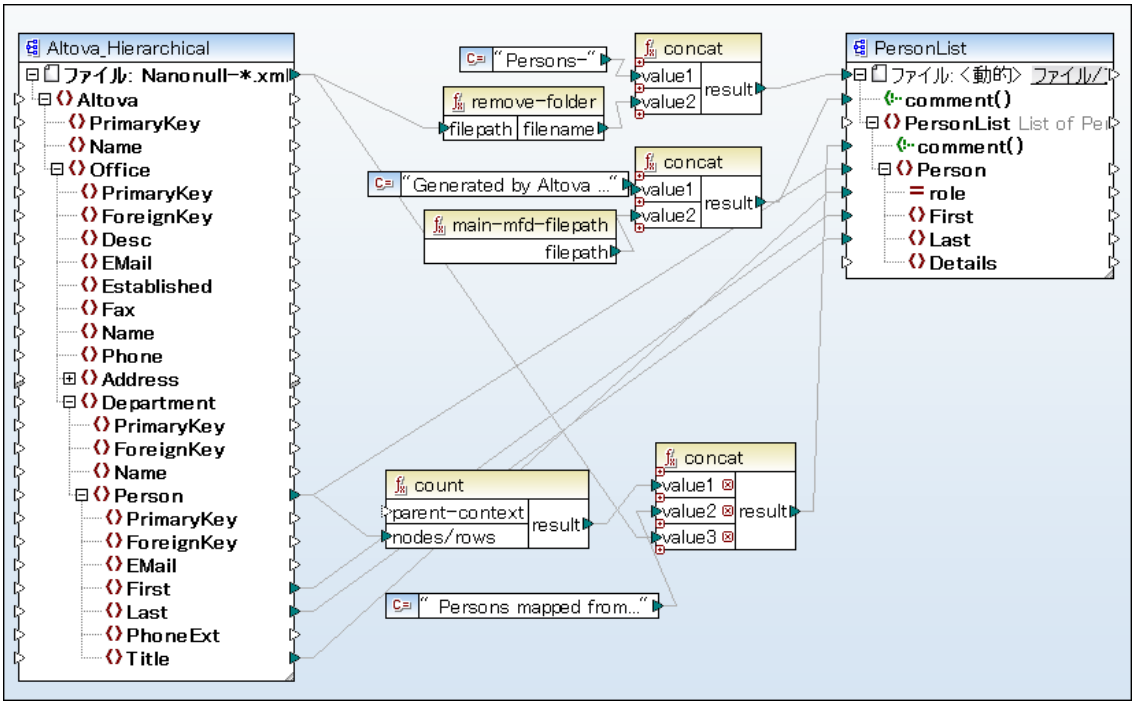


MergeMultipleFiles_List.mfd (MapForce Basic Edition)

5.4.2 複数の入力ファイルを複数の出力ファイルにマッピングする

複数のファイルを複数の ターゲットファイルにマップするには、一意の出力ファイル名を生成する必要があります。出力ファイル名が入力データ内の文字列から派生する場合があり、他の場合、入力ファイル名から出力ファイル名を派生させることが役にたつ場合があります。例：ファイル拡張子の変更により。

以下のマッピングでは、**concat** 関数を使用してプレフィックス "Persons-" を追加し、出力ファイル名を入力ファイル名から派生させます。



MultipleInputToMultiple 出力」Files.mfd (MapForce Basic Edition)

メモ: 処理関数を使用することなく、入力と出力ルートノードを直接接続することを回避します。これを行うことによりマッピングが実行される際、入力ファイルが上書きされます。上記の**concat** 関数などを使用して、出力ファイル名を変更することができます。

メニューオプション **「ファイル| マッピング設定」** によりマッピングで使用されるファイルパス設定をグローバルに定義することができます（マッピング設定の変更を参照）。

5.4.3 ファイル名をマッピングパラメーターとして提供する

マッピングにカスタムファイル名を入力パラメーターとして提供するには、以下を行います：

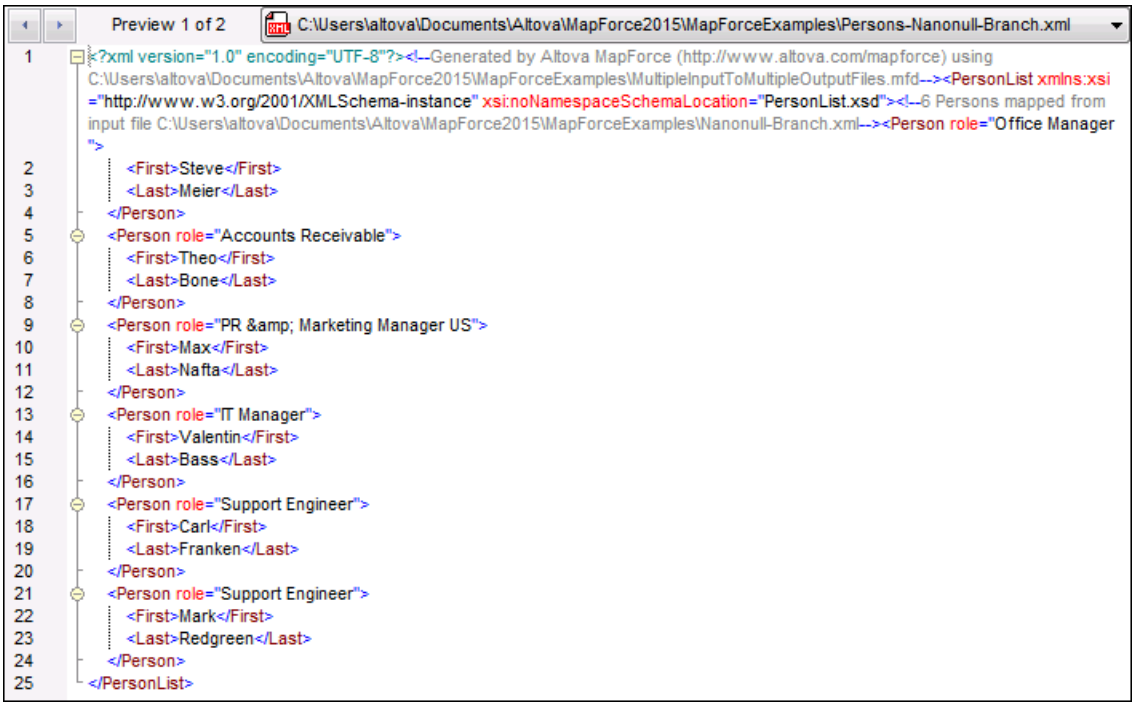
1. 入力コンポーネントをマッピングに追加する（**「関数」** メニューから **「入力の挿入」** をクリックします）。コンポーネントについての更に詳しい情報は、単純型入力を参照。
1. ソースコンポーネントの **「ファイル」**（File）または **「ファイル文字列」**（File/String）ボタンをクリックし、**「マッピングから与えられた動的なファイル名を使用する」** を選択します
2. マッピングソースとして振舞うコンポーネントのルートノードに入力パラメーターを接続します。

成功例は例：名前をマッピングパラメーターとして使用するを参照してください）。

5.4.4 複数の出力ファイルをプレビューする

プレビューウィンドウにマッピングの結果を表示するために、「出力」タブをクリックします。マッピングが複数の出力ファイルを作成する場合、各ファイルは、「出力」タブ内でペインごとに番号が付けられます。矢印をクリックして、それぞれの出力ファ

イルを確認してください。



MultipleInputToMultipleOutputFiles.mfd

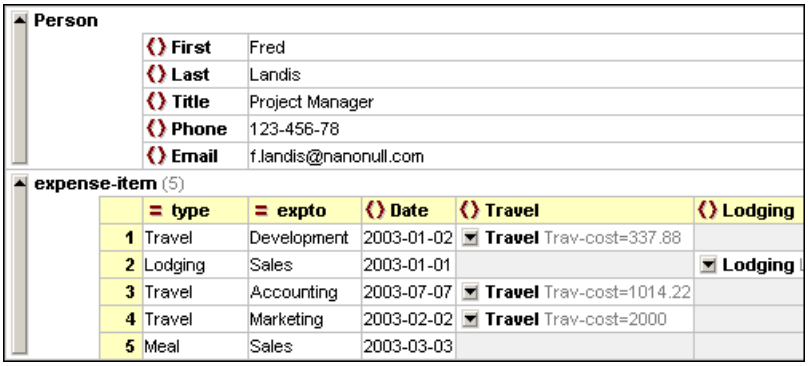
生成された出力ファイルを保存するためには以下を行います：

- **出力**」メニューから**Save**「**すべての出力ファイルを保存**」（ ）をクリックします。
- **すべての生成された出力を保存**」（ ）ツールバーボタンをクリックします。

5.4.5 サンプル：1つの XML ファイルを複数のファイルに分割

このサンプルは、単一のソースXML ファイルから動的に複数のXML ファイルを作成する方法を説明します。このサンプルに使用されるマッピングは以下で検索することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Tut-ExpReport-dyn.mfd**。

ソースXML ファイル（マッピングと同じフォルダーで見つけることのできる）は、"Fred Landis" という個人の経費レポートから構成されており、5つの異なる種類の経費アイテムを含んでいます。このサンプルの目的は個々の経費アイテムのためにXML ファイルを生成することです。

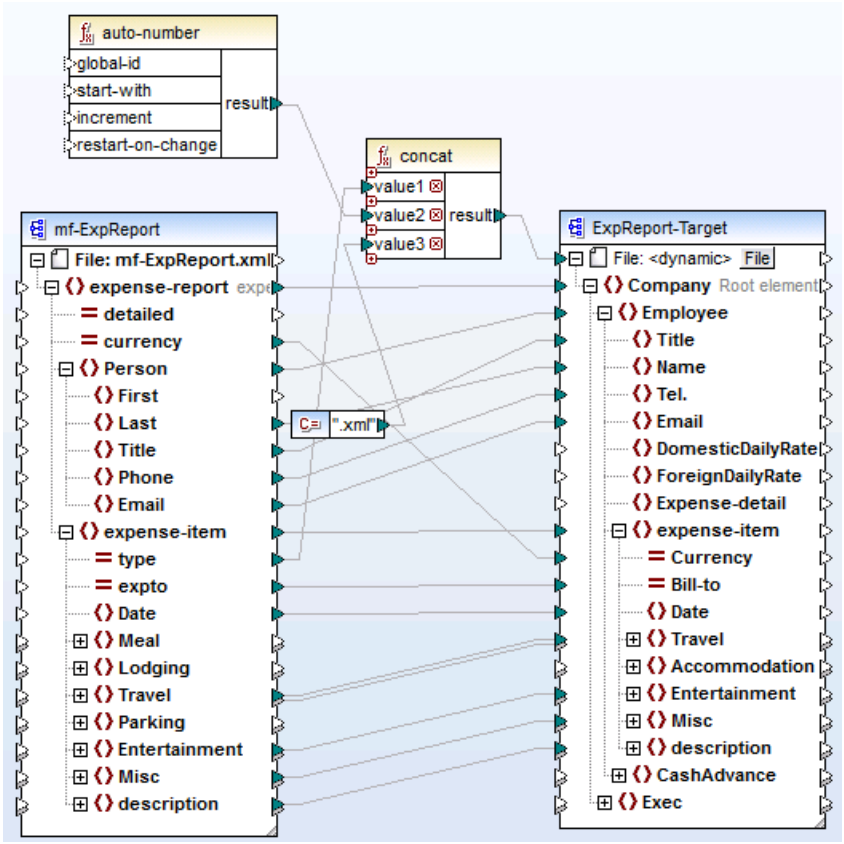


Person					
First	Fred				
Last	Landis				
Title	Project Manager				
Phone	123-456-78				
Email	f.landis@nanonull.com				
expense-item (5)					
	type	expto	Date	Travel	Lodging
1	Travel	Development	2003-01-02	Travel Trav-cost=337.88	
2	Lodging	Sales	2003-01-01		Lodging
3	Travel	Accounting	2003-07-07	Travel Trav-cost=1014.22	
4	Travel	Marketing	2003-02-02	Travel Trav-cost=2000	
5	Meal	Sales	2003-03-03		

mf-ExpReport.xml (XMLSpy グリッドビューにて表示)

`type` 属性は特定の経費アイテムの型を定義し、これはソースファイルを分割するために使用されるアイテムの型です。このエクササイズのゴールを達成するには、以下を行います :

1. (『ライブラリ』ペインの **『core | string 関数』** ライブラリからドラッグして)**`concat`** 関数を挿入します。
2. 定数を入力するには、(**『挿入』** メニューから **『定数』** をクリックして) "xml" を値として入力します。
3. (『ライブラリ』ペインの **『core | generator 関数』** からドラッグして)**`auto-number`** 関数を入力します。
1. **『ファイル』**(File) または ターゲットコンポーネントの **『ファイル文字列』**(File/String) ボタンをクリックし、**『マッピングから与えられた動的なファイル名を使用する』** を選択します
4. 下に表示されるとおり接続を作成し、**『出力』** タブをクリックしマッピングの結果を確認します。



Tut-ExpReport-dyn.mfd (MapForce Basic Edition)

結果的出力ファイルは、以下のとおり動的に名前が付けられます：

- `type` 属性は、ファイル名の最初の部分を提供します。（例："Travel"）。
- **自動連番** 関数は、ファイルの連番を提供します。（例："Travel1"、"Travel2"、など）。
- 定数は "xml" であるファイル拡張子を提供します。このため "Travel1.xml" は最初のファイル名です。

5.4.6 サンプル：データベーステーブルに対して複数の XML を生成

この例は、データベーステーブルのそれぞれのレコードに対して、動的に複数の XML ファイルを生成する方法を説明します。この例は、以下で参照することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\PersonDB-dyn.mfd**。

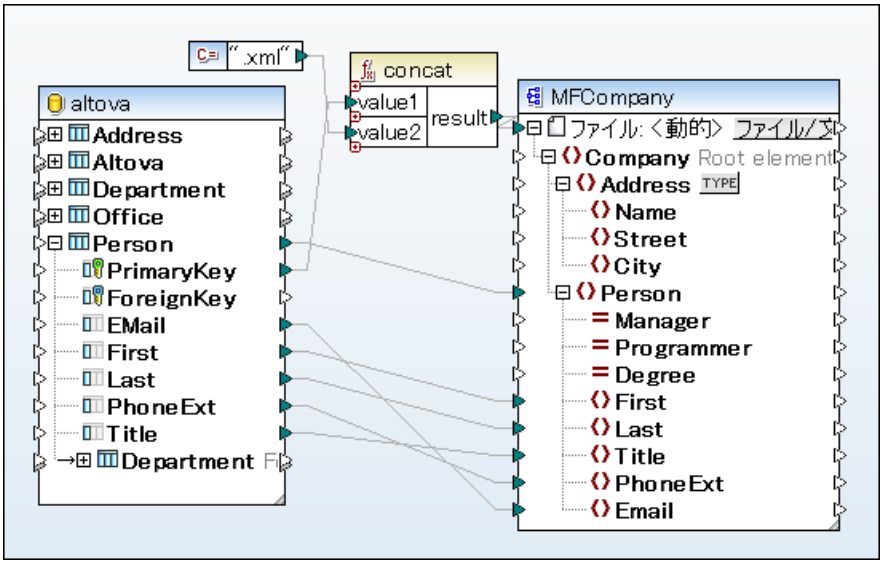
ソースデータベースファイルには 5 つのテーブルがあり、Person テーブルには 21 名の人物に関するレコードが収められています。Person テーブル内のそれぞれのレコードに対し、個別の XML ファイルを生成することがこのサンプルの目的です。

Person

PrimaryKey	ForeignKey	EMail	First	Last
1	1	v.callaby@nanonu	Vernon	Callaby
2	1	f.further@nanonu	Frank	Further
3	1	l.matise@nanonu	Loby	Matise
4	2	j.firstbread@nanc	Joe	Firstbread
5	2	s.sanna@nanonul	Susi	Sanna
6	3	f.landis@nanonul	Fred	Landis
7	3	m.landis@nanonu	Michelle	Butler
8	3	t.little@nanonull.	Ted	Little

"PrimaryKey" フィールドの内容にてテーブル内の人物を（データベースでは）識別しているため、このアイテムを使用してソースデータベースにあるデータを複数のファイルへ分割します。このエクササイズのゴールを達成するには、以下を行います：

1. `concat` 関数を挿入します。（『ライブラリ』ペインの **「コア」文字列関数」** ライブラリから関数をドラッグすることができます）。
2. 定数を入力するには、（**「挿入」** メニューから **「定数」** をクリックし）"xml" を値として入力します。
3. **「ファイル」**（File）または「ターゲットコンポーネントの **「ファイル文字列」**（File/String）ボタンをクリックし、**「マッピングから与えられた動的なファイル名を使用する」** を選択します
4. 下で表示されるように接続を作成します。**「出力」** タブをクリックしてマッピングの結果を確認します。



PersonDB-dyn.mfd (MapForce Professional Edition)

結果的出力ファイルは、以下のとおり動的に名前が付けられます：

- 「PrimaryKey」フィールドは、名前の最初の部分を提供します（例："1"）。
- 定数は、ファイル拡張子（".xml"）を提供します。このため "1.xml" は最初のファイルのファイル名です。

5.4.7 Excel の行から複数の XML ファイルを生成

..\MapForceExamples\Tutorial フォルダー内に収められている**altova.xlsx** 表計算ファイルを以下に示します。10行のデータが含まれているAdmin ならびに1行のデータが含まれているDevelopment という2つのワークシートから構成されています。以下にある例は ..\Tutorial フォルダーの**Excel-Mapping-dyn.mfd** にて確認することができます。

Admin ワークシート

	A	B	C	D
1	Valentin	Bass	716	v.bass@nanon
2	Theo	Bone	331	t.bone@nanon
3	Vernon	Callaby	582	v.callaby@nan
4	Joe	Firstbread	621	j.firstbread@na
5	Frank	Further	471	f.further@nano
6	Alex	Martin	778	a.martin@nano
7	Loby	Matise	963	l.matise@nano
8	Steve	Meier	114	s.meier@nano
9	Max	Nafta	122	m.nafta@nano
10	Susi	Sanna	753	s.sanna@nano
11				
12				

Admin | Development

Ready

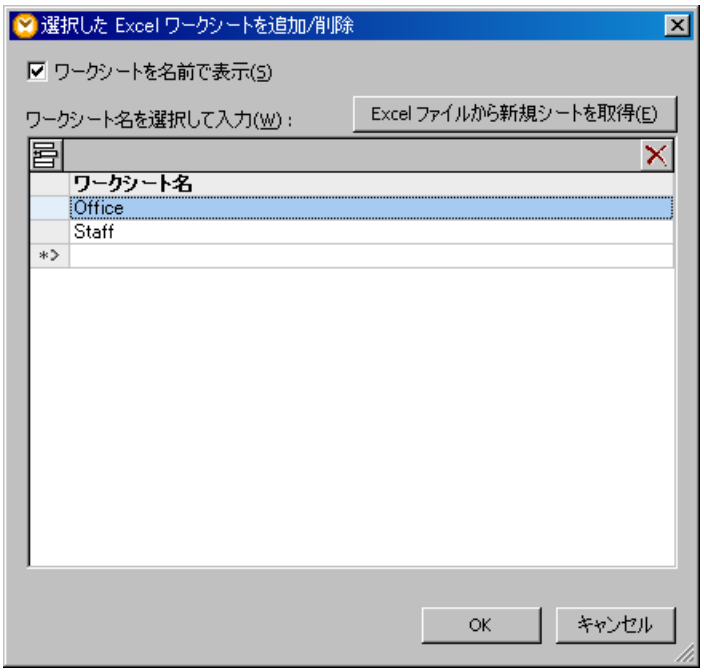
Development ワークシート

	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@nar
7	Fred	Landis	951	f.landis@n
8	Ted	Little	852	t.little@nai
9	Mark	Redgreen	152	m.redgreer
10	Paul	Smith	334	p.smith@r
11	Ann	Way	951	a.way@na
12				

Admin | Development

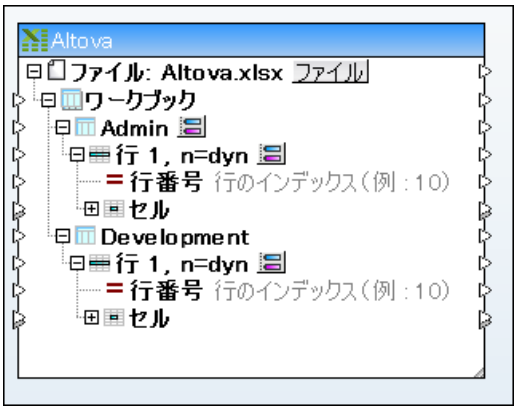
Ready

MapForce ではコンポーネントオプションで指定された方法に従い Excel コンポーネントの表示ならびにマッピングを行うことができます。デフォルトの設定を以下に表示します。Excel コンポーネントを挿入した際には、「ワークシートを名前で表示」チェックボックスが有効になります。

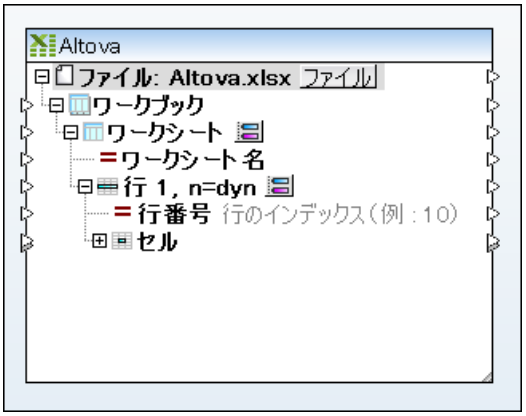


単一のワークシートとしてワークブックへアクセスする：

1. Excel コンポーネントのAdmin の隣にある アイコンをクリックします。



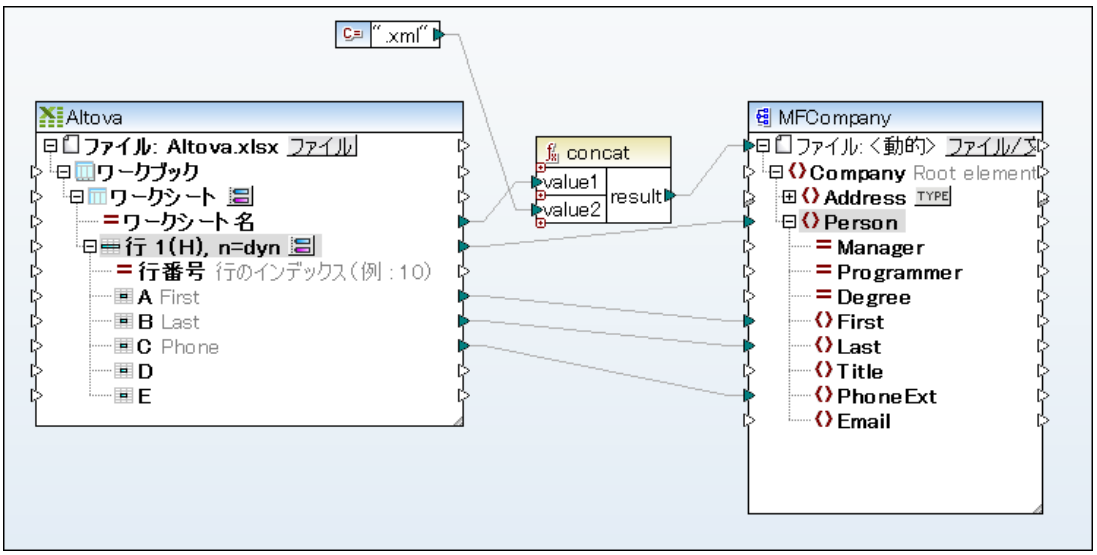
2. 「ワークシートを名前で表示」チェックボックスをクリックして、チェックを外します。
以下のスクリーンショットに示されるように、ワークシートを名前で検索することができなくなりましたが、ワークシート名アイテムが使用できるようになりました



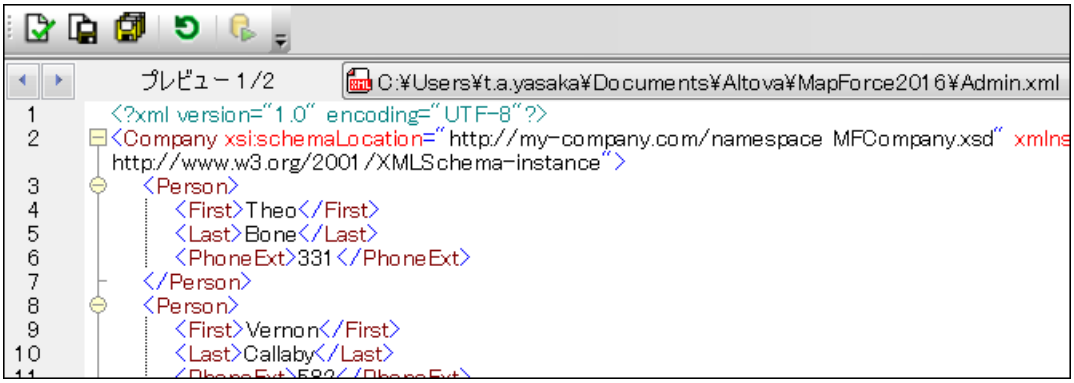
目的 1:人物に関する情報を含む各ワークシートに対して別々のファイルを生成します。

「ワークシート名」アイテムにより、ワークブックにある特定のワークシートが決定されるため、このアイテムを使用することでソースとなるワークブックを複数のファイルへ分割することができます。

1. ライブラリペインからconcat と定数関数を挿入します。



2. 上で示されているような接続を作成します。サークシート名をvalue1 へ接続し、定数をvalue2 へ接続します。
3. concat 関数の**結果** 出力を、ターゲットコンポーネントの**File:** アイテムへ接続します。ファイルアイテムが ファイル : <dynamic> と表示されることに注目してください
4. その他の接続を必要に応じて作成します。
5. 出力タブをクリックして、マッピングの結果を確認します。



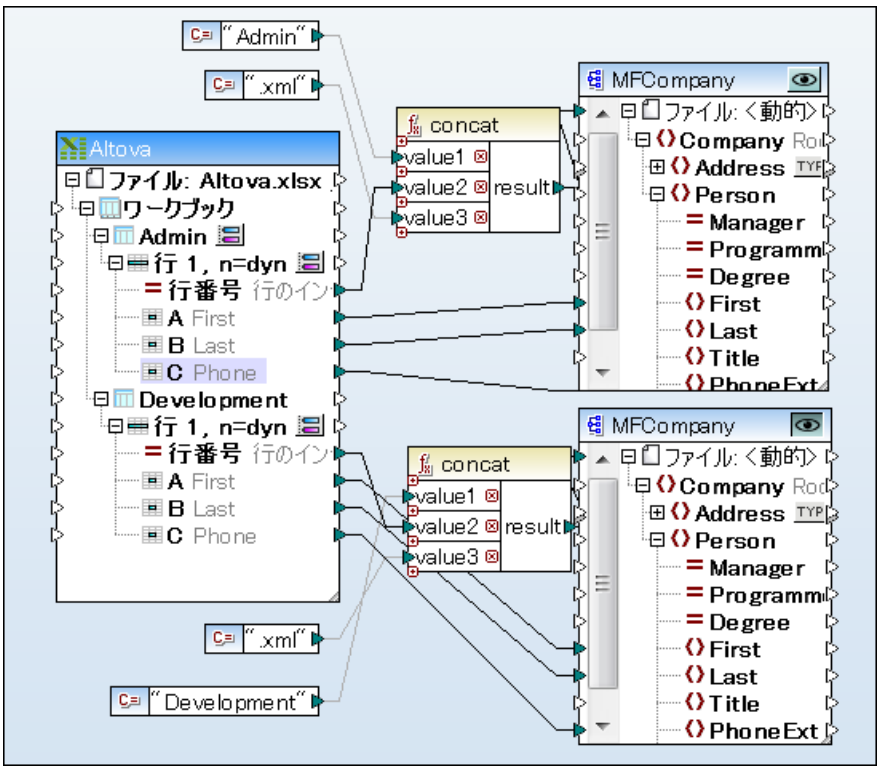
Excel ワークシートにある各行がプレビュータブに表示されます。上のスクリーンショットは最初のプレビューを表示したものです。次へ/前へ 矢印をクリックすることで、出力タブにある各ファイルを確認することができます。

メモ:

- **ワークシート**名フィールドは、ファイル名における最初の部分（語頭）となります。例 :Admin.
- **定数**コンポーネントにより、ファイル拡張子の**.xml** が与えられ、最初のファイルは Admin.xml となります。Admin.xml にはその Excel タブにある全ての行が含まれます。Development.xml には、2番目のワークシートにある行が含まれます。
- 全て保存アイコン をクリックすることで、コードの生成を行うこと無く全てのファイルを出力タブから直接保存することができます。

目的 2: ワークシート内に記述された各人物ごとに別々のファイルを生成する。

1. 上記の手順を踏んだ後で、ワークシートの隣にある アイコンをクリックし、「ワークシートを名前で表示」チェックボックスにチェックを入れます。Admin とDevelopment の各ワークシートがコンポーネントにて表示されます。

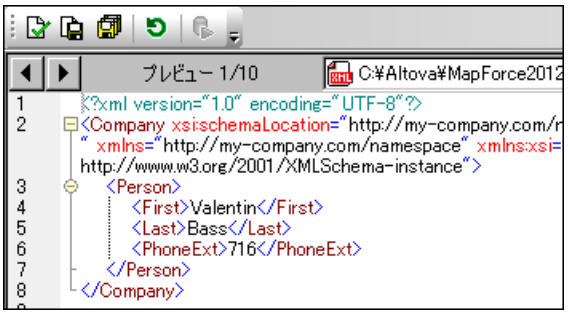


2. 上に示されるように、定数コンポーネントとconcat コンポーネントを挿入します。
3. 同じ名前で 2番目のXML スキーマファイルを挿入し、上に示されるような接続を作成します。

   行番行の属性によりワークシートにある人物の行が決定されるため、このアイテムを使って、各ワークシートを別々のファイルへ分割します。

Altova XLSX コンポーネントにある**Admin** ワークシート:

1. 上に示されるような接続を作成します。Admin 定数をvalue1 へ **うえ**をvalue2 へ そして "xml" 定数をvalue3 へ。
2. concat 関数の**result** 出力を、ターゲットコンポーネントの**ファイル** アイテムへ接続します。ファイルアイテムが ファイル: <dynamic> と表示されることに注目してください。
3. その他の接続を必要に応じて作成します。
4. 出力タブをクリックして、マッピングの結果を確認します。



各行の変換結果をプレビュータブにて確認することができます。Admin ワークシート内にある10のレコード全てが別々のファイルに分割されました。次へ 前へ 矢印をクリックすることで、出力タブ内にある各ファイルを確認することができます。

メモ:

- Admin **定数**がファイル名の最初の部分を構成します。例 :Admin.
- **行番号**アイテムにより、Excel ワークシートの行番号（例：1）が与えられます。
- ".xml" **定数**によりファイルが保存される際の拡張子名が与えられ、結果としてAdmin1.xml というファイル名にてデータが保存されます。
- 全て保存アイコン により、コードを生成することなく、出力タブから全てのファイルを保存することができます。

Development ワークシートの出力を確認するには、Development に対応したターゲットコンポーネントのプレビューボタンをクリックし、出力ボタンをクリックすることで、11 レコードのマッピング結果を確認することができます。

5.5 マッピングにパラメーターを与える

単純型入力コンポーネントを介して、単純型の値をマッピングにパスすることができます。マッピングエリアでは、単純型入力コンポーネントは、この結果、アイテムとシーケンスの構造の代わりに、単純型のデータ型を持つソースコンポーネントの役割を果たします (例 : 文字列、整数など)。 ファイルベースの ソースコンポーネントの代わりに または追加して 単純型入力コンポーネントを作成することができます

以下のMapForce 変換言語で単純型入力コンポーネントを使用することができます。:

- BUILT-IN (**「プレビュー」**タブからMapForce 内で直接マッピングの変換をプレビューする場合)
- BUILT-IN (コンパイルされたMapForce Server 実行ファイルを実行する場合)
- XSLT 1.0、XSLT 2.0
- XQuery
- C++
- C#
- Java

MapForce Server を使用してマッピングを実行、または生成されたコードを使用して実行する場合、単純型入力コンポーネントは、コマンドライン パラメーターになります。 マッピングがXSLT 変換として生成される場合、単純型入力コンポーネントは、生成されたXSLT ファイル内のスタイルシートパラメーターに対応します。

それぞれの単純型入力コンポーネント (または パラメーター) をオプションまたは必須として作成することができます (入力コンポーネント設定を参照)。必要であれば、マッピング入力パラメーターのためにデフォルトの値を作成することができます (デフォルトの入力値を作成するを参照)。これによりマッピングの実行時にパラメーターの値を明示的には提供しない場合でも、マッピングを安全に実行することができます。

マッピングエリアに追加された入力パラメーターをユーザー定義関数内の入力パラメーターと混同しないように注意してください。 (ユーザー定義関数を参照)。類似点と相違点は、以下のとおりです。

マッピング上の入力パラメーター	ユーザー定義関数の入力パラメーター
**関数	入力の挿入」**メニューから追加する。	**関数	入力の挿入」**メニューから追加する。
単純型データ型を持つことができます (文字列、整数など)。	単純型と複雑型のデータ型を持つことができます
マッピング全体に適用可能。	定義された方法で関数のコンテキストに適用することができます。

逆のマッピングを作成する場合、 (メニューコマンド **「ツール | 逆順マッピングの作成」**を使用すると単純型入力コンポーネントは、単純型出力コンポーネントになります。

サンプル : 例 : 名前をマッピングパラメーターとして使用するを参照。

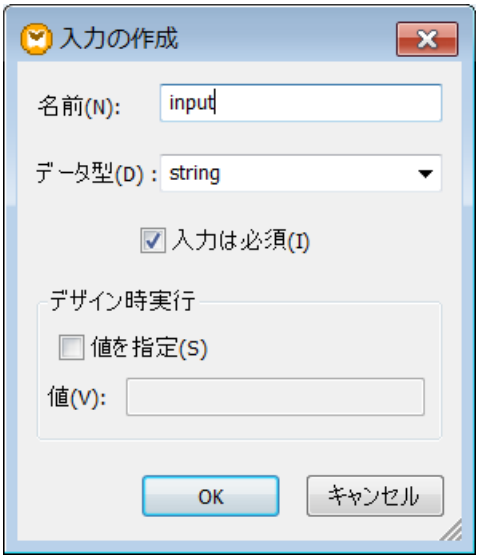
5.5.1 単純型入力コンポーネントの追加

マッピングに単純型入力コンポーネントを追加する :

1. マッピング ウィンドウに (ユーザー定義関数ではない) メインマッピングが表示されていることを確認してください。
2. **「関数」**メニューから **「入力」**をクリックします。
3. 名前を入力して、この入力に必要なデータ型を選択します。入力が必須のマッピングパラメーターとして扱われる必要がある場合、 **「入力は必須」**チェックボックスを選択します。設定の完全なリストは、単純型入力コンポーネント

設定を参照してください。

4. 「OK」をクリックします。

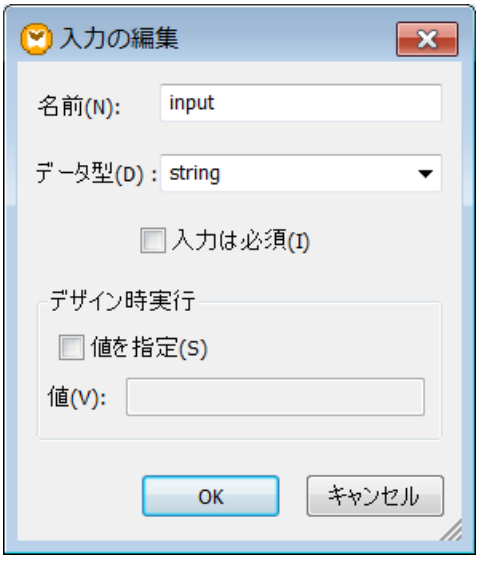


入力の作成 ダイアログボックス

ここで定義されている設定は後に変更することができます（単純型入力コンポーネント設定を参照）。

5.5.2 単純型入力コンポーネント設定

単純型入力コンポーネントに適用することのできる設定をマッピングエリアに追加する際に定義することができます。後に設定を「入力の編集」ダイアログボックスから変更することも可能です。



入力の編集 ダイアログボックス

「入力の編集」ダイアログボックスを開くには、以下を行います：

- コンポーネントを選択して、「コンポーネント」メニューから「プロパティ」をクリックします。

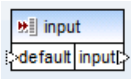
- コンポーネントをダブルクリックします。
- コンポーネントを右クリックして、 **「プロパティ」** をクリックします。

使用することのできる設定は、以下のとおりです：

名前	このコンポーネントに対応する入力パラメーターのための詳細名を入力します。マッピングの実行時、このテキストボックスに入力された値がマッピングに与えられるパラメーターの名前になります。ですから、スペースまたは特別な文字は許可されていません。
データ型	デフォルトでは、全ての入力パラメーターは文字列データ型として扱われます。パラメーターが異なる型を持つ場合、リストからそれぞれの値を選択します。マッピングが実行された場合、MapForce は、入力パラメーターをここで選択されたデータ型にキャストします。
入力は必須	有効化されている場合、この設定は、入力パラメーターを必須にします（すなわち、パラメーターの値が提供されない限り、マッピングを実行することはできません）。 入力パラメーターのためのデフォルトの値を指定するには、このチェックボックスを無効化にします（デフォルトの入力値を作成するを参照）。
値の指定	この設定は、デザイン時にマッピングを実行する場合のみ適用することができます。 **「プレビュー」** タブをクリックすることにより、コンポーネント内に直接マッピング入力として使用される値を入力することができます。
値	この設定は、デザイン時にマッピングを実行する場合のみ適用することができます。 **「プレビュー」** タブをクリックすることにより、マッピング入力として MapForce で使用される値を入力するには、値の指定チェックボックスを選択し、必要な値を入力します。

5.5.3 デフォルトの入力値を作成する

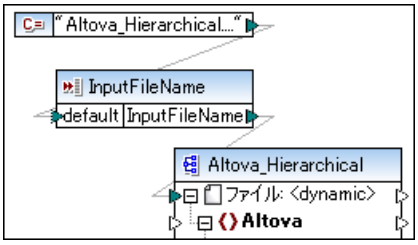
入力コンポーネントをマッピングエリアに追加すると、**default** アイテムがコンポーネントの左側に表示されます。



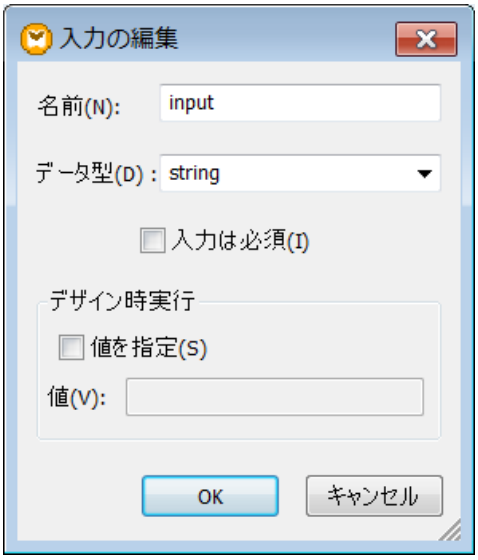
単純型入力コンポーネント

デフォルトアイテムは任意のデフォルトの値のこの入力コンポーネントへの接続を以下のように有効化します：

1. 定数コンポーネントを追加し、（ **「挿入」** メニューから **「定数」** をクリックします。）入力コンポーネントの **default** アイテムに接続します。



2. 入力コンポーネントをダブルクリックして、**「入力は必須」** チェックボックスのチェックを外します。デフォルトの入力の値を作成すると、この設定は重要になり、マッピングの検証に関する警告を引き起こす可能性があります。



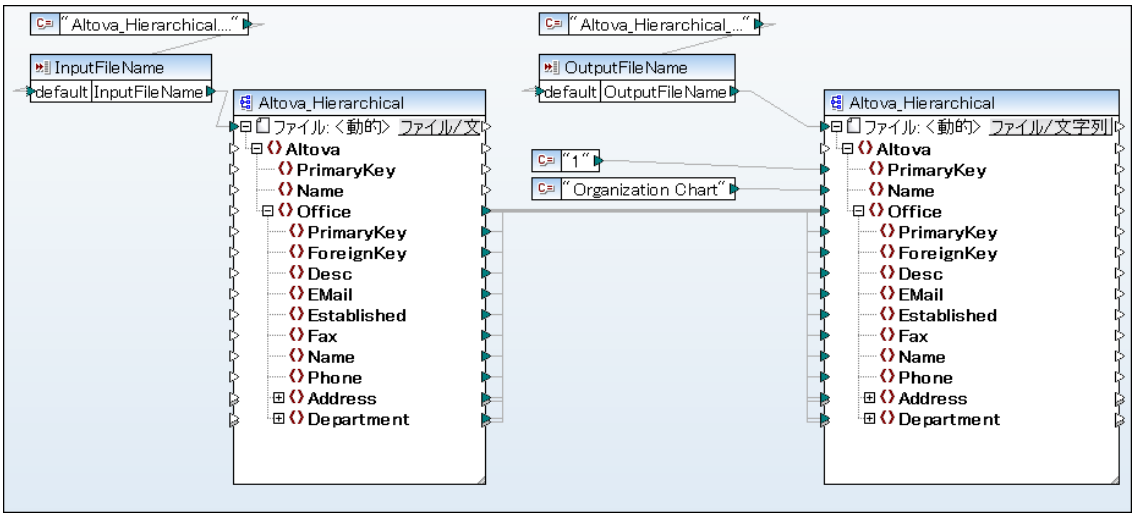
3. 「**OK**」をクリックします。

メモ: 値の指定 チェックボックスをクリックし横のボックスに値を入力した場合、入力された値は、マッピングをプレビューする際に、以前入力された（すなわち、デザイン時の実行時の）デフォルトの値を上書きします。しかしながら、同じ値は生成されたコードでは効果をもたらしません。

5.5.4 例：名前をマッピングパラメーターとして使用する

このサンプルはランタイムに入力パラメーターを取り込むマッピングの実行に必要なステップを説明します。マッピングデザインファイルは、以下のパスにあるサンプルを使用します：**<Documents>\Altova\MapForce2017\MapForceExamples\FileNamesAsParameters.mfd**.

マッピングは2つの入力コンポーネントを使用します：**InputFileName** と**OutputFileName**。これらのコンポーネントはソースとターゲットXMLファイルの入力ファイル名（および出力ファイル名をそれぞれ）与えます。この理由のため、これらのコンポーネントは、**File: <dynamic>** アイテムに接続されています。



FileNamesAsParameters.mfd (MapForce Basic Edition)

InputFileName と**OutputFileName** コンポーネントは、マッピング内の単純型入力コンポーネントですので、マッピングを実行する際に入力パラメーターとして提供することができます。以下のセクションは以下の変換言語での実行方法を説明しています：

- XSLT 2.0 RaptorXML Server を使用
- ビルトイン (MapForce Server 実行ファイル)MapForce Server を使用
- Java

XSLT 2.0

XSLT 1.0 または XSLT 2.0 出コードを生成すると、入力パラメーターは、RaptorXML Server での実行のため **DoTransform.bat** バッチファイルに書き込まれます（RaptorXML Server を参照）。**DoTransform.bat** ファイルを呼び出す際、異なった入力（または出力）ファイルを使用するため、コマンドラインで必要なパラメーターをパスするか、または、必要なパラメーターを含むように後者を編集します。

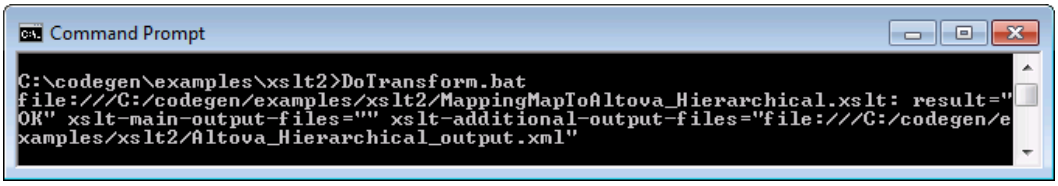
DoTransform.bat ファイル内でカスタム入力パラメーターを提供する：

1. **FileNamesAsParameters.mfd** サンプルから XSLT 2.0 コードを生成（**「ファイル | コード生成 | XSLT 2.0」**）します。
2. **<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリから **Altova_Hierarchical.xml** ファイルを XSLT 2.0 コードが直接生成されるディレクトリにコピーします（このサンプルでは、**c:\codegen\examples\xslt2**）。このファイルは、カスタムパラメーターとしての役割を果たします。
3. **DoTransform.bat** をカスタム入力パラメーターが（下でハイライトされているように）%* の前または後に含まれるように編集します。パラメーターの値は一重引用符で閉じられていることに注意してください。使用することのできる入力パラメーターは **rem**（リマーク）セクションにリストされています。

```
@echo off

RaptorXML xslt --xslt-version=2 --
input="MappingMapToAltova_Hierarchical.xslt" --
param=InputFileName:'Altova_Hierarchical.xml' %*
"MappingMapToAltova_Hierarchical.xslt"
rem --param=InputFileName:
rem --param=OutputFileName:
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

DoTransform.bat ファイルを実行すると、RaptorXML Server は **Altova_Hierarchical.xml** を入力パラメーターとして使用して、変換を完了します。



MapForce Server 実行ファイル

カスタム入力パラメーターを MapForce Server 実行ファイルは提供するには、以下を行います：

1. **FileNamesAsParameters.mfd** を MapForce Server 実行ファイルにコンパイルします（MapForce マッピングのコンパイルを参照）。プロンプトされると、mfx 実行ファイルを使用中のコンピューターのディレクトリに保存します。（このサンプルでは、**c:\codegen\examples\mfx**）。
2. **<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリから **Altova_Hierarchical.xml** ファイルを .mfx ファイルを保存した場所にコピーします。このファイルはマッピングの

実行ファイルは提供されたカスタムパラメーターとしての役割を果たします。

3. MapForce Server を以下のコマンドで実行する:

```
MapForceServer.exe run "C:\codegen\examples\mfx
\FileNamesAsParameters.mfx" -p=InputFileName:"C:\codegen\examples\mfx
\Altova_Hierarchical.xml" -p=OutputFileName:"C:\codegen\examples\mfx
\OutputFile.xml"
```

上のMapForce Server コマンドでは、`-p=InputFileName` と `-p=OutputFileName` は、マッピングに対する入力パラメーターです。**-OutputFileName** の値として全てのファイル名を使用することができます。しかしながら、**-InputFileName** パラメーター内で提供されるファイル名は物理ファイルとして存在する必要があります。それ以外の場合、マッピングは失敗します。

メモ: l"MapForceServer.exe は、内部コマンドまたは外部コマンド、操作可能なプログラムまたはバッチ ファイルとして認識されていません。" というメッセージが表示された場合、MapForce Server 実行可能ファイルがインストールされている現在のディレクトリに変更してください。マッピングを実行する都度パスを変更することを回避するために、オペレーティングシステムのPATH 環境変数にMapForce Server 実行可能ファイルがインストールされているディレクトリのパスを追加してください (例 :**C:\Program Files (x86)\Altova\MapForceServer2017 \bin**) 。

Java

カスタム入力パラメーターをJava .jar アプリケーションに提供する場合 :

1. Java コードを**FileNamesAsParameters.mfd** サンプルから生成します (**「ファイル|コードの生成 | Java」**) 。
2. Java コードを実行可能なJAR ファイルにコンパイルします。 (Eclipse 内での命令に関しては、例 :Eclipse とAnt を使用してJava アプリケーションを構築する を参照) 。
3. **Altova_Hierarchical.xml** ファイルを**<Documents>\Altova\MapForce2017 \MapForceExamples** ディレクトリから jar ファイルが存在する場所にコピーします。Java マッピングアプリケーションに提供されたカスタムパラメーターとして振る舞います。
4. コマンドラインで以下を入力します :`java -jar Mapping.jar /InputFileName "InputFile.xml"`

.jar ファイルにパラメーターをパスする際にワイルドカードを使用する場合、ワイルドカードのパラメーターを引用符内に置きます。例 :

```
java -jar Mapping.jar /InputFileName "altova-*.xml"
```

5.6 マッピングから文字列の値を返す

マッピングから文字列を返すことが必要な場合単純型出力コンポーネントを使用します。マッピングエリアでは、単純型出力コンポーネントは、アイテムとシーケンスの構造の代わりに文字列データ型を持つターゲットコンポーネントの役割をします。この結果、単純型出力コンポーネント（または追加して）ファイルベースのターゲットコンポーネントの代わりに作成することができます。. 例：単純型出力コンポーネントを使用して、素早く関数の出力をテストとプレビューすることができます関数の出力（例：関数の出力をテストするを参照）。このテクニックは文字列のシリアル化を使用するマッピングでも役にたちます（文字列の解析とシリアル化を参照）。単純型出力コンポーネントの主な目的は、しかしながら、MapForce Server API を呼び出す際にファイルに書き込むことなく文字列に戻ることです。

単純型出力 コンポーネントをユーザー定義関数の出力パラメーターと混同しないように注意してください。（ユーザー定義関数を参照）。類似点と相違点は、以下のとおりです。

出力コンポーネント	ユーザー定義関数出力パラメーター
**関数	出力の挿入**」メニューから追加	**関数	出力の挿入**」メニューから追加
文字列をデータ型として持ちます。	単純型と複雑型のデータ型を持つことができます
マッピング全体に適用することができます。	定義された方法で関数のコンテキストに適用することができます。

必要であれば、複数の単純型出力 コンポーネントをマッピングに追加することができます。単純型出力 コンポーネントをファイルベースのターゲットコンポーネント とデータベース と共に使用することができます。マッピングが複数のターゲットコンポーネントを含む場合、特定のコンポーネントにより返されるデータをプレビューすることができます。コンポーネントタイトルバー内の**「プレビュー」**（ ）ボタンをクリックすることにより、マッピングウィンドウの **「出力」**タブをクリックします。

単純型出力 コンポーネントをMapForce 変換言語内で以下のように使用することができます：

言語	作動のしくみ
BUILT-IN（マッピング変換のプレビュー時）	ファイルベースの マッピング出力をプレビューする同様、マッピングウィンドウの **「出力」** タブをクリックすることにより、出力コンポーネントをプレビューすることができます。
BUILT-IN（MapForce Server 実行ファイルの実行時）	コンパイルされたMapForce Server 実行ファイルを実行すると（MapForce のマッピングをコンパイルを参照）、マッピング出力は、標準出力 ストリーム (stdout) 内に返され、表示またはファイルへリダイレクトすることができます。例：MapForce server 実行ファイルの名前が**MyMapping.mfx** とします。use 以下の構文を使用して、マッピング出力を**output.txt** ファイルにリダイレクトし、全てのエラーを**log.txt** ファイルにリダイレクトします： `MapForceServer.exe run MyMapping.mfx >output.txt 2>log.txt`
XSLT 1.0、XSLT 2.0	マッピング内で定義された単純型出力コンポーネント生成されたXSLT ファイルが変換の出力になった場合。 RaptorXML Server を使用する場合、RaptorXML Server にマッピング出力をファイルに書き込み、`--output` パラメータに値としてパスするように命令することができます。 ファイルに出力を書き込む、**DoTransform.bat** ファイル内の`--output` パ

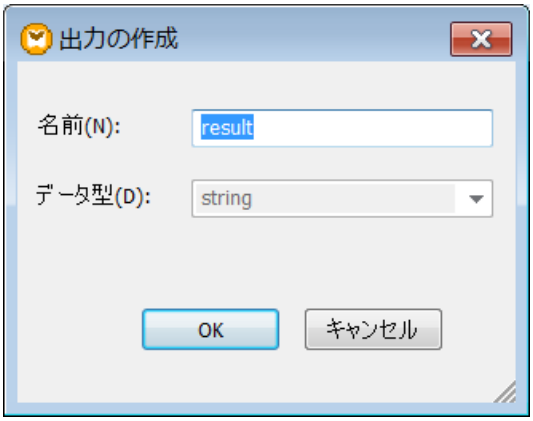
	ラメーターに追加または編集する場合。例：**次のDoTransform.bat** ファイルは**Output.txt** ファイルへのマッピング出力に書き込むために編集されました（ハイライトされたテキスト参照してください）。 `RaptorXML xslt --xslt-version=2 --input="MappingMapToResult1.xslt" --output="Output.txt" %* "MappingMapToResult1.xslt"` `--output` パラメーターが定義されていない場合、マッピングが実行される際、マッピング出力が標準出力ストリーム（stdout）に書き込まれます。
C++、C#、Java	生成されたC++、C#、とJava コード内で、マッピング出力は、生成されたアプリケーションの標準出力に書き込まれます。 マッピングが複数のターゲットコンポーネントを含む場合、生成されたアプリケーションは、各ターゲットコンポーネント標準出力を連結して、1つのまとめられた標準出力を返します。

逆のマッピングを作成する場合、（メニューコマンド **「ツール | 逆マッピングの作成」** を使用すると）単純型出力コンポーネントは、単純型入力コンポーネントになります。

5.6.1 単純型出力コンポーネントの追加

マッピングエリアに出力コンポーネントを追加する：

1. ウィンドウが（ユーザー定義関数ではない）インのマッピングを表示していることを確認してください。
2. **「関数」** メニューから **「出力」** をクリックします。
3. コンポーネントのための名前を入力します。
4. **「OK」** をクリックします。



出力の作成ダイアログボックス

コンポーネント名を以下の方法で後に変更することができます：

- コンポーネントを選択して、 **「コンポーネント」** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。

5.6.2 サンプル :関数出力のプレビュー

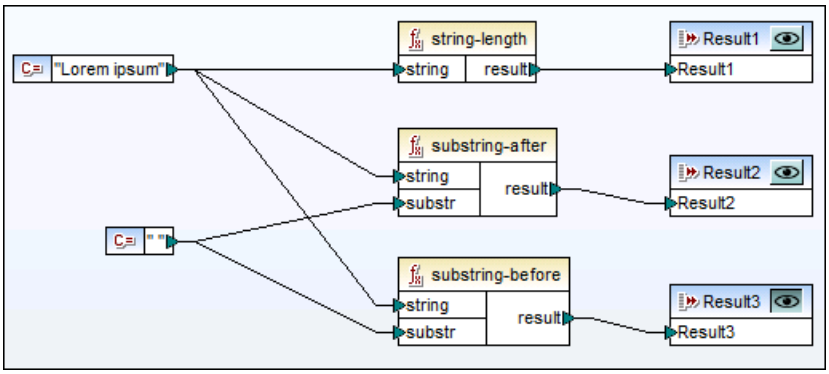
この例は、単純型出力 コンポーネントを使用して MapForce 関数により返された出力を確認する方法を説明します。関数に関する基本的理解と MapForce 関数 に関する全般的な知識があると、このサンプルをよく理解することができます。MapForce 関数の基礎知識がない場合、例を開始する前に 関数の使用 を参照してください。

この例の目的は、マッピングエリアに複数の関数を追加することで、また単純型出力コンポーネントを使用して出力をプレビューする方法を理解することです。特に、例はcore ライブラリ内で使用することのできる複数の関数をマッピングエリアに追加する方法を説明しています。次に使用方法の概要が述べられています :

string-length	は、引数として与えられた文字列内の文字の数を返します。例 :これを関数値 "Lorem ipsum" にパスすると、テキスト "Lorem ipsum" が持つ文字の数である結果は "11" です。
substring-after	は、引数として与えられたセパレーターの後に発生する文字列の一部を返します。例 :これを関数値 "Lorem ipsum" とスペース文字 (" ") にパスすると、結果は "ipsum" です。
substring-before	は、引数として与えられたセパレーターの前に発生する文字列の一部を返します。例 :これを関数値 "Lorem ipsum" とスペース文字 (" ") にパスすると、結果は "Lorem" です。

カスタムテキストの値に対して関数をテストするには (このサンプルでは "Lorem ipsum") 以下のステップを行います :

1. 値を持つ定数 "Lorem ipsum" マッピングエリアを追加します (メニューコマンド **挿入 | 定数** を使用して)。定数はテストされる関数のそれぞれの入力パラメーターになります。
2. **string-length**、**substring-after** と**substring-before** 関数をcore ライブラリ **string 関数** セクションからマッピングエリアにドラッグして、マッピングエリアに追加します。
3. 空のスペース (" ") を値として持つ定数を追加します。これは**substring-after** と**substring-before** 関数により必要とされるセパレーターパラメーターとなります。
4. 単純型出力 コンポーネントを (メニューコマンド **関数 | 出力の挿入** を使用して) 追加します。他の名前を与えることも可能ですが、このサンプルでは、*Result1*、*Result2*、および *Result3*, と名前が付けられています。
5. コンポーネントを下に表示されるように接続してください。



単純型出力 コンポーネントを使用して関数の出力をテストする

上のサンプルで示されるとおり、"Lorem ipsum" 文字列はそれぞれの、**string-length**、**substring-after**、および**substring-before** 関数入力パラメーターとして振る舞います。更に、**substring-after** と **substring-before** 関数は、スペースの値を第 2入力パラメーター として取ります。*Result1*、*Result2*、および *Result3* コンポーネントは各関数の結果をプレビューするために使用することができます。

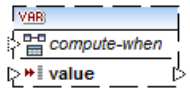
関数の出力をプレビューする

- コンポーネントタイトルバー内の **「プレビュー」**（ ）ボタンをクリックし、マッピングウィンドウの **「出力」** タブをクリックします。

5.7 変数の使用

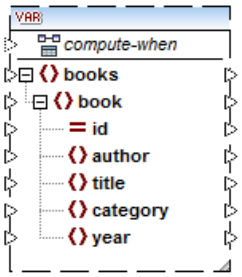
変数とは更なる処理のために中間マッピングの結果を保管するために使用される特別なコンポーネントです。マッピング上のデータを一時的に「記憶」し、処理する必要がある場合があります。例えば、ターゲットコンポーネントにコピーされる前に、フィルター、または、関数を適用するなど。

変数は、単純型（例えば、文字列、整数、ブール値、など）または 複合型（ツリー構造）であることができます。



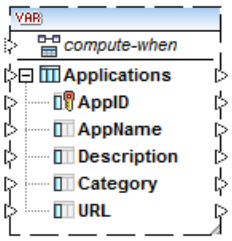
単純型変数

変数の構造を示す XML スキーマを提供して複合型の変数を作成することができます。スキーマが要素をグローバルに定義すると、変数構造のルートノードになるものを選択することができます。変数に関連した インスタンス XML ファイルが存在しない場合、変数のデータはマッピングランタイムに計算されます。



XML スキーマから作成された複合型変数

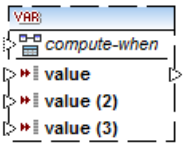
データベース、EDI ファイル または FlexText テンプレート（後者についての更に詳しい情報に関しては、次を参照してください：MapForce FlexText）から複合型の変数を作成することも可能です。 データベースの場合は、変数構造のためのルートアイテムとして特定のデータベーステーブルを選択することができます。



データベーステーブルから作成された複合型変数

上のイメージでは、各変数には、`compute-when` という名前のアイテムが存在することに注意してください。このアイテムに接続することは任意です。これにより、マッピング上でどのように変数の値が計算されるかを管理することができます（次を参照してください：変数のコンテキストとスコープの変更）。

必要な場合、通常のコンポーネントと同様に、変数構造のアイテムを1つ以上のソース接続から受け入れられるように複製することができます（次を参照してください：入力の複製）。これは、しかしながらデータベーステーブルから作成された変数には適用されません。



複製された入力を持つ単純型変数

変数に関して最も重要な点は、変数はシーケンスであり、シーケンスを作成するために使用されるという点です。「シーケンス」という用語は、ここでは、ゼロまたは、それ以上のアイテムのリストを意味します（次も参照してください：マッピングのルールと戦略）。これにより、変数はマッピングの寿命内で複数のアイテムを処理することができるようになります。しかしながら、値を変数に与え、マッピングのその他の部分を同じに保つことも可能です（次を参照してください：変数のコンテキストとスコープの変更）。

ある程度までは、変数をチェーンマッピングの中間コンポーネントと比較することができます（次を参照してください：チェーンマッピング）。しかしながら、マッピング内の各段階で中間ファイルを作成する必要がない場合、変数は柔軟性があり便利です。次のテーブルの概要は変数とチェーンマッピングの違いを示しています。

チェーンマッピング	変数
チェーンマッピングには、2つの独立したステップが含まれます。例えば、A、B、とC の3つのコンポーネントがマッピングに存在するとします。マッピングの実行には2つの段階があります：A からB へのマッピングの実行、および、B からC へのマッピングの実行。	マッピングの実行中、変数は、コンテキストとスコープに従い評価されます。コンテキストとスコープを影響することはできます（次を参照してください：変数のコンテキストとスコープの変更）。
マッピングが実行されると、中間結果は外部でファイルに保管されます。	マッピングが実行されると、中間の結果は内部で保管されます。変数の結果を含む外部ファイルは作成されません。
中間 結果は ボタンを使用してプレビューすることができます。	マッピング ランタイムで計算される変数の結果をプレビューすることはできません（しかしながら、デバッガーを使用して、変数により生成されたデータを確認することができます（次を参照してください：マッピングのでデバッグ）。

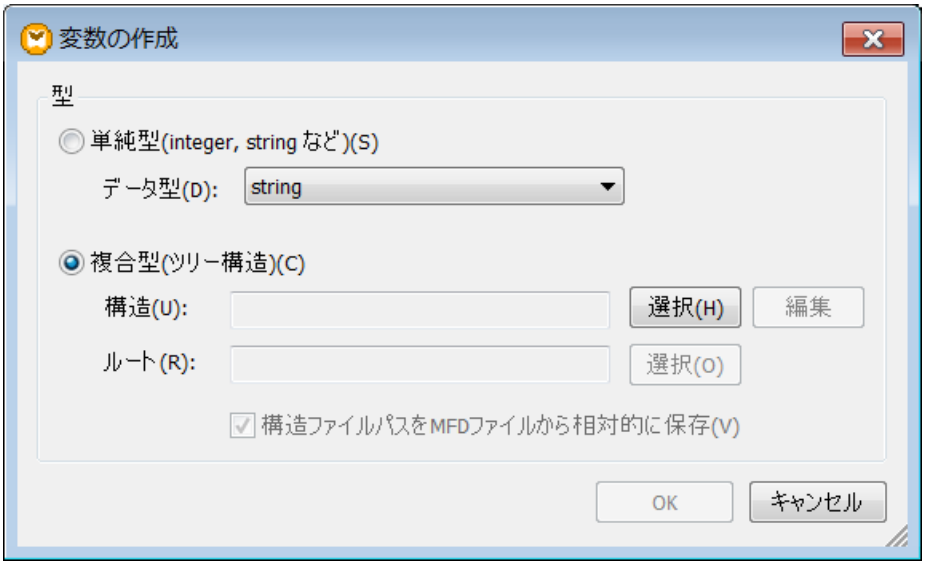
メモ： 変数は、マッピング変換言語がXSLT 1.0 に設定されていない場合サポートされません。

5.7.1 変数の追加

マッピングに変数を追加するには、以下に示されているようにいくつかの方法があります。

メニュー、または、ツールバーコマンドの使用

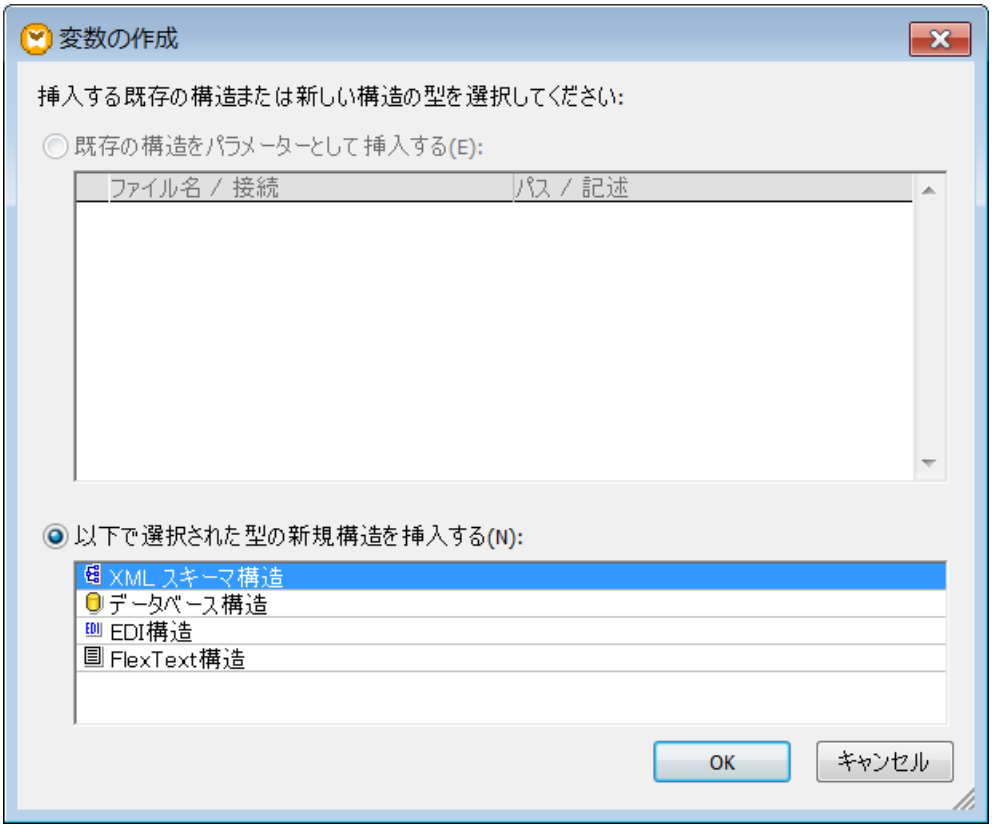
1. **挿入**」メニューから、**変数**」をクリックします。（**変数** ツールバーボタンを代わりにクリックします）。



2. 挿入する変数の型を選択します (単純型 または 複合型)。

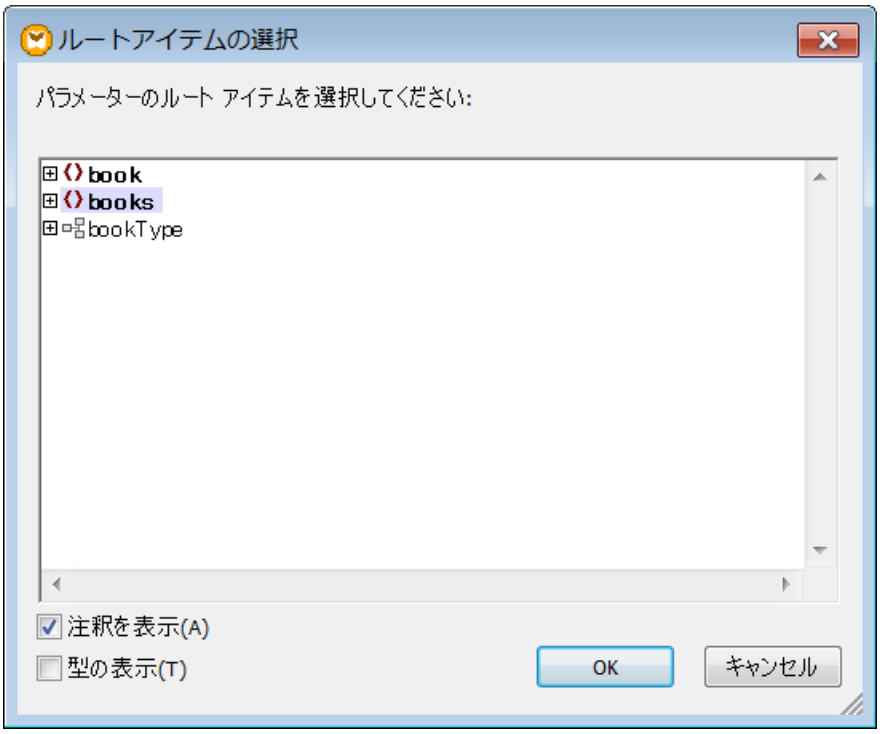
"複合型 "を選択した場合、追加ステップが存在します :

3. 変数の構造を与えるソースを選択するために、 **選択** をクリックします (例えば、XML スキーマ、 データベース、EDI ファイル、 または FlexText テンプレート)。



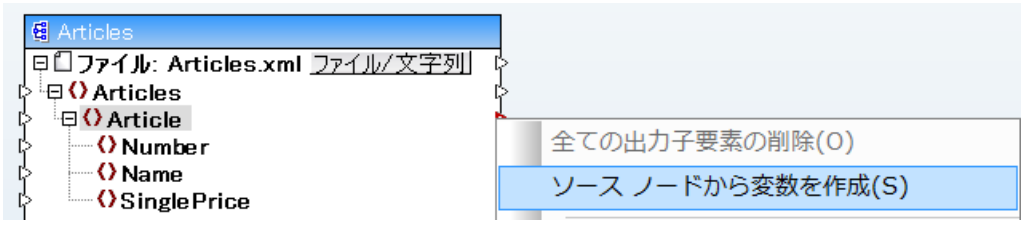
4. プロンプトされると、構造のルートアイテムを指定します。XML スキーマの場合、ルート アイテムは、グローバルに定義されている すべての要素であることができます。データベースの場合は、ルートアイテムはすべてのテーブル であるこ

とができます。

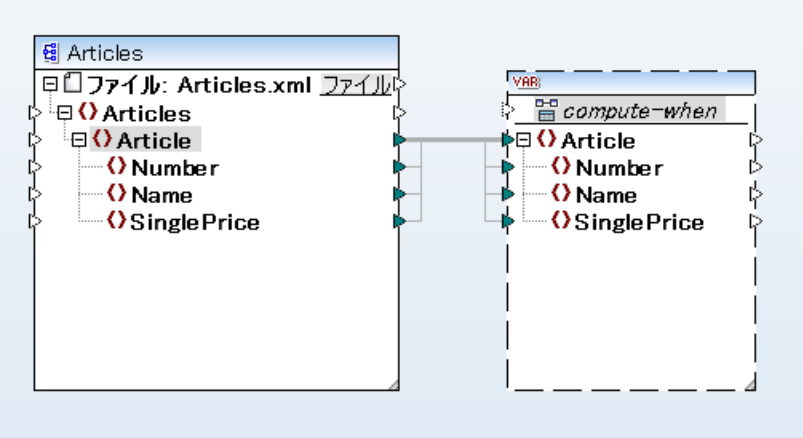


コンテキストメニューの使用

- コンポーネントの出力コネクタを右クリックします (このサンプルでは、 Article」) そして、**「ソースノードから変数を作成する」**を選択します。



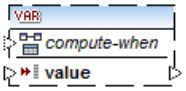
これにより、同じソーススキーマを使用する複合型の変数を作成し、自動的にすべてのアイテムをすべてをコピーする接続により接続します。



- ターゲットコンポーネントの入力コネクタを右クリックして、**「ターゲットノードのために変数を作成する」**を選択します。これにより、同じスキーマをターゲットとして使用し、複合型の変数を作成します。そして、すべてコピー接続を使用してすべてのアイテムを自動的に接続します。
- フィルターコンポーネント (on-true/on-false) の出力コネクタを右クリックして、**「ソースノードから変数を作成する」**を選択します。これにより、ソーススキーマを使用して複合型コンポーネントを作成し、フィルター入力にリンクされているアイテムを中間コンポーネントのルート要素として自動的に使用します。

5.7.2 変数のコンテキストとスコープの変更

各変数には、変数のスコープを管理することを許可するcompute-when 入力アイテムが存在します。すなわち、マッピングが実行される際に変数の値がいつ、どの頻度で計算されるかを管理することができます。この入力は、多くの場合接続する必要がありませんが、デフォルトのコンテキストを上書きする場合、または、マッピングのパフォーマンスを最適化するには、必要です。



「compute-when」アイテム

次のテキストでの、サブツリーは、ターゲットコンポーネント内のアイテム/ノードセットとその子孫を意味します。例えば、<FirstName> と<LastName> 子要素を持つ <Person> 要素。

変数の値 は、変数コンポーネントの出力サイドでデータが使用できることを意味します。

- 単純型の変数に関しては、コンポーネントプロパティ内で指定されているデータ型を持つ動的な値のシーケンスであることを意味します。
- 複合型の変数に関しては、それぞれが自身の子孫ノードを含む (コンポーネント プロパティ内で指定されている型の) ルートノードのシーケンスであることを意味します。

動的な値 (または、ノード) のシーケンスは、1つの要素、または、要素を全く含まない場合があります。これは、変数の入力サイドに何が接続されているか、および、ソースとターゲットコンポーネント内の親アイテムの存在により異なります。

「compute-when」が接続されていない場合 (デフォルト)

compute-when 入力アイテムが (ソースコンポーネントの出力ノードに) 接続されていない場合、ターゲットサブツリー内で最初に使用される際に変数の値は (コネクタにより変数コンポーネントからターゲットコンポーネント内のノードに直接、または、関数を使用して間接的に) 計算されます。同じ変数の値は、サブツリー内のターゲット子ノードすべてのために使用されます。

実際の変数の値は、ソースとターゲットコンポーネントの親アイテム間の接続により異なります。

このデフォルトの振る舞いは、正規ユーザー定義 関数とWeb サービス関数の呼び出しの複合型の出力と同じです。

変数の出力が複数の関連したにターゲットノードに接続されている場合、変数の値は、それぞれのアイテムのために個別に計算されます。これにより、各ケース内で異なる結果を生成することができます。これは異なる親の接続は、変数の値が評価されるコンテキストを影響するからです。

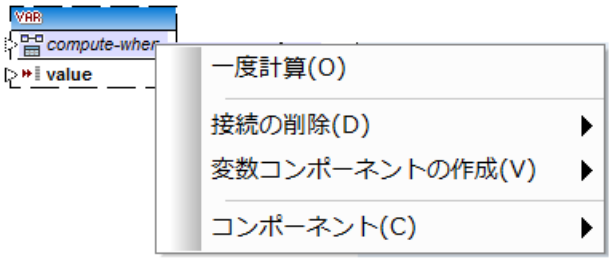
「compute-when」が接続されている場合

ソースコンポーネントの出力コネクタを`compute-when` に接続すると、ソースアイテムが最初にターゲットサブツリー内で使用される都度に変数が計算されます。

変数は実際には、`compute-when` に接続されているアイテムの子アイテムのように振る舞います。これは、新規のアイテムがソースコンポーネント内のシーケンスから読み込まれる都度、ランタイム変数が再評価されます。これは、MapForce 内の接続を管理する一般的なルールに関連しています：各ソースアイテムのために、1つのターゲットアイテムが作成されます。`compute-when` に関しては、各ソースアイテムのために、変数の値が計算されます (次を参照してください：マッピングのルールと戦略)。

「compute-once」

必要であれば、ターゲットコンポーネントの前に1度変数の値を計算し、変数をマッピングの残りでグローバルな定数にすることを選択することができます。これをおこなうには、`compute-when` アイテムを右クリックして、コンテキストメニューから **「1度計算する」** を選択します：



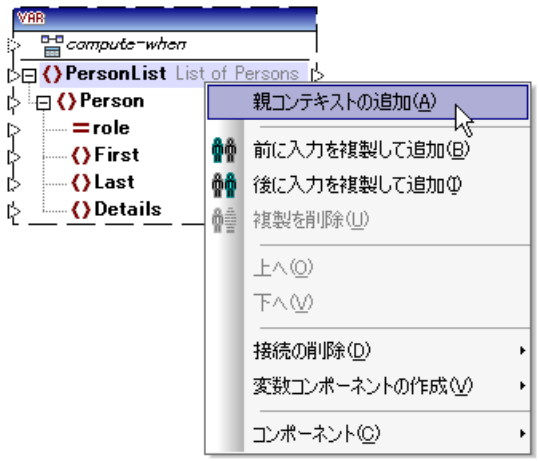
変数のスコープを`compute-when=once` に変更する場合、このような変数は1度のみ評価されるため、入力コネクタは、`compute-when` アイテムから削除されます。

実際の関数の結果が評価される前に、ユーザー定義関数 `compute-when=once` 変数が関数が呼び出される都度評価されます。

親コンテキスト

親コンテキストを追加する必要がある場合があります。例えば、マッピングが複数のフィルターを使用し、反復するために親ノードを追加する必要がある場合など。 マッピングコンテキストのオーバーライド を参照してください。

変数に親コンテキストを追加するには、ルートノード (このサンプルでは 「PersonList」) を右クリックします、そして、コンテキストメニューから、 **親コンテキストを追加する** を選択します。これにより新規ノード、親コンテキストを既存の階層構造に追加します。

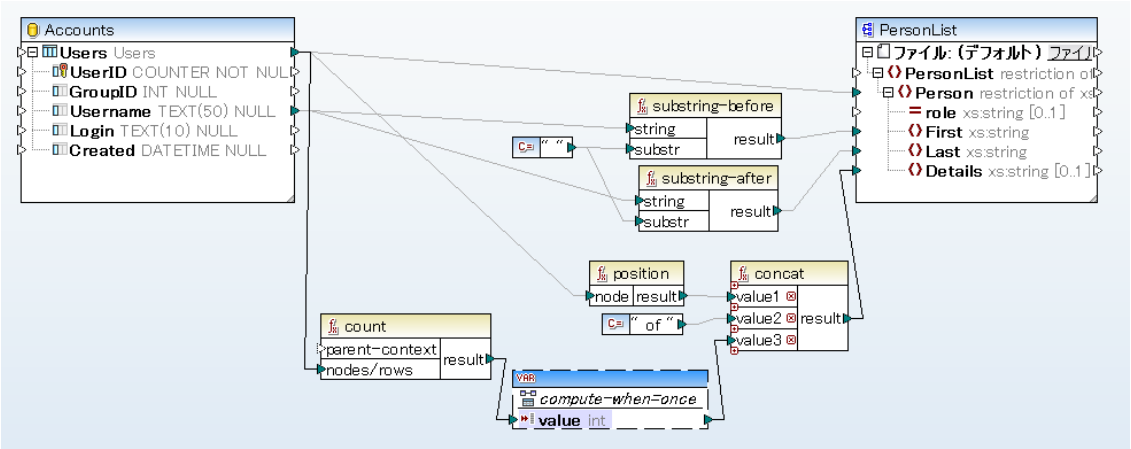


親コンテキストは、仮想の「親」ノードをコンポーネント内の階層構造に追加します。これにより、同じ、または、異なるソースコンポーネント内で追加ノードを反復することができます。

5.7.3 サンプル :データベースのテーブルの行の計算

このサンプルで説明されているマッピングは、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の**DB_UserList.mfd** です。このマッピングは、ユーザーのレコードを "Users" という名前のデータベーステーブルから抽出し、XML ファイルに書き込みます。データベースの列 "Username" には、個人の姓と名の両方が含まれています (例えば、"Vernon Callaby")。このマッピングの目的は以下のとおりです :

1. "Users" テーブル内の各レコードのために、XML ファイル内に新規の`Person` 要素を作成する。
2. データベースフィールド "Username" から抽出された値をXML ファイル内の個別のフィールド ("First" と "Last") に分割する。
3. 各レコードのために、データベース内に存在するレコードの全体の番号に対して比較される連番を検索し (例えば、"4 件中の 1件目 "など) `Details` 要素にこの情報を書き込みます。



DB_UserList.mfd

上で説明されているとおり、最初の目的を達成するために、ソース "Users" テーブルとターゲットXML ファイルの`Person` 要素 の間に接続が描かれます。これにより、ソーステーブル内の各レコードのために、ターゲット内に新規の`Person` 要素が作成されます。

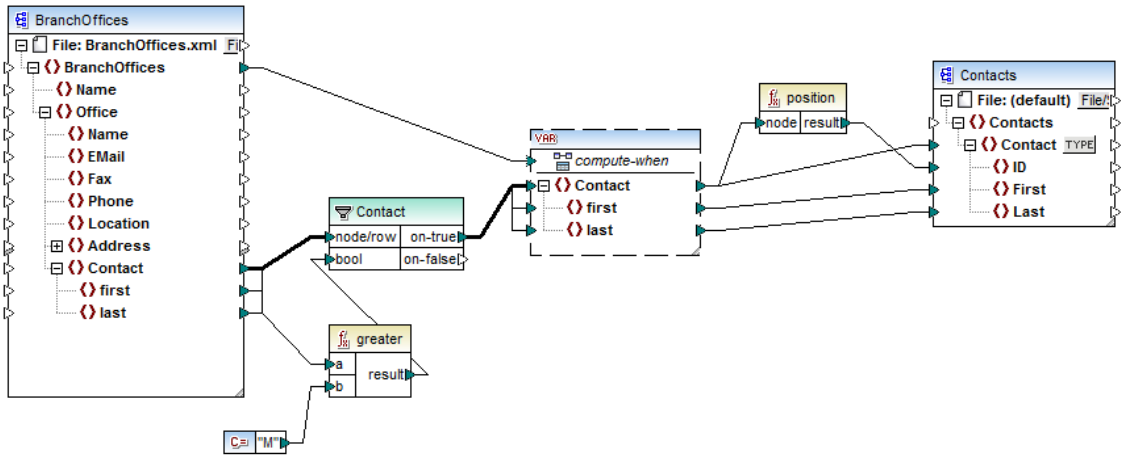
フィールド "Username" の値が**substring-before** と**substring-after** 関数に与えられています。これらの2つの関数は、スペース文字 (" ")の前後のテキストをそれぞれ抽出し、このマッピングの目的を達成します。

3つ目の目的を達成するために、マッピングは**count** 関数を使用します。集計関数の結果は、変数にパスされます。変数は、マッピング上に結果が保管され、各個人の "Details" 要素 をターゲットXML に書き込む際に使用することができます。効率性のために、データベースのレコードは1度のみ数えられるべきであり、変数のスコープはcompute-when=once に設定されています (次を参照してください:変数のコンテキストとスコープの変更)。

5.7.4 サンプル :データベースのテーブルの行の計算

このサンプル内で説明されているマッピングのシーケンスは、**<Documents>\Altova\MapForce2017 \MapForceExamples** フォルダー内の**PositionInFiltered.mfd** で使用することができます。

このマッピングは、複数の個人の連絡先のデータを含むXML ファイルを読み取り、フィルターし、ターゲットXML ファイルに書き込みます。このマッピングの目的は、ソースXML ファイルから、「M」または、以降のアルファベットから始まる姓を持つ個人をフィルターします。抽出された連絡先には、番号がつけられている必要があります。番号は、ターゲットXML ファイル内で各連絡先の一意の識別子としての役割を果たします。



PositionInFilteredSequence.mfd

上の目的を達成するには、次のコンポーネント型 がマッピングに追加されました:

- フィルター (次を参照してください:フィルターと条件)
- 複合型変数 (次を参照してください:変数の追加)
- 関数 **greater** と**position** (次を参照してください:関数との作業)
- 定数 (定数を追加するには、メニューコマンド **挿入 | 定数** を選択します)。

変数は、ソースコンポーネントと同じスキーマを使用します。変数を右クリックして、コンテキストメニューから**プロパティ**を選択すると、この変数構造のためにルートノードとして ノード**BranchOffices/Office/Contact** が選択されていることを確認してください。

最初に、ソースコンポーネントのデータは、フィルターにパスされます。フィルターは、変数にフィルターの条件を満たすレコードのみをパスします。具体的には、フィルターは、最初の名前が「M」に等しい、または、大きい値を持つContact ノードのみを取得するように構成されています。この目的を達成するには、関数 **greater** は、各 last アイテムを定数値 「M」と比較します。

変数には、ソースコンポーネント (branchOffices) のルートアイテムに接続されているcompute-when 入力があります

す。ランタイムでは、これは、ソースコンポーネント内のシーケンスから新規のアイテムが読み込まれる都度、変数を再評価するようにします。このマッピングでは、しかしながら、compute-when アイテムの接続、または、未接続に違いはありません。この理由は、変数が(フィルターを使用して間接的に) Contact ソースアイテムに接続されており、フィルター条件を満たす Contact のインスタンスの数だけ計算がおこなわれるからです。

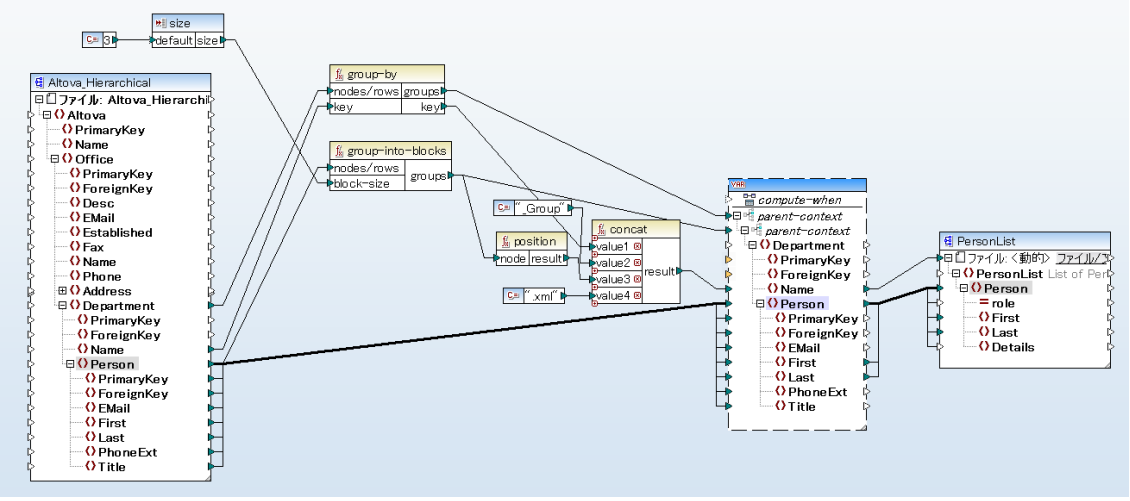
position 関数は、変数の各反復のために現在のシーケンスの番号を返します。8件の連絡先がフィルターの条件を満たしています。ですから、マッピングをプレビューして、出力を確認すると、1から8のID が、ターゲットコンポーネントのID 要素に書き込まれていることが確認することができます。

変数の必要性に関しては、すべてのレコードに番号付けをする必要性に応じるものです。フィルターの結果を直接ターゲットコンポーネントに接続すると、Contact の各発生に番号を付けることができません。このマッピング内の変数の目的は、ですから、Contact の各インスタンスを一時的にマッピングに保管し、ターゲットに書き込まれる前に番号を付けることです。

5.7.5 サンプル 記録のグループ化、および、サブグループ化

このサンプルで説明されているマッピングは、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の**DividePersonsByDepartmentIntoGroups.mfd** です。

このマッピングは、架空の会社の社員のレコードを含むXML ファイルを処理します。社内には2つのオフィスが存在します：「Nanonull, Inc.」と「Nanonull Partners, Inc」。各オフィスには複数の部署が存在します (例えば、「IT」、「マーケティング」など)。また、各部署には一名以上の社員が存在します。マッピングの目的は、オフィスに関わりなく各部署から3人までで構成されているグループを作成することです。グループのサイズはデフォルトでは3名です。しかしながら、必要に応じて変更することもできます。各グループは、フォーマット "<Department Name>_GroupN" の名前を持つ個別のXML ファイルとして保存される必要があります (例えば、**Marketing_Group1.xml**、**Marketing_Group2.xml**、など)。



DividePersonsByDepartmentIntoGroups.mfd

上で説明されているとおり、マッピングの目的を達成するには、複合型変数と住に関数である)他の コンポーネント型をマッピングに追加します。ソースXML 内のDepartment アイテムと同じ構造が変数にも存在します。プロパティを表示するために変数を右クリックすると、ソースコンポーネントと同じXML スキーマが使用されていること、および、ルート要素として Department が存在していることに気が付きます。重要な点は、各部署のコンテキスト内で変数が最初に計算され、次に各グループのコンテキスト内で計算されるように、変数には、2つのネストされた親コンテキストアイテムが存在することです (次も参照してください：変数のコンテキストとスコープの変更)。

始めに、マッピングは、各部署の名前を取得するためにすべての部署を反復します (これは各グループに対応するファイル名を後に作成するために必要とされます)。これは、**group-by** 関数をDepartment ソースアイテムに接続すること、部署

名をグループ化のキーとして与えることにより達成されます。

次に、各部署のコンテキスト内で、2番目のグループ化が発生します。具体的には必要とされる一のグループを作成するために、マッピングが**group-into-blocks** 関数を呼び出します。デフォルトの値が "3" である単純型のコンポーネントが各グループのサイズを提供します。デフォルトの値は、定数により与えられます。グループのサイズを変更するために、このサンプルでは、必要に応じて定数を簡単に変更することができます。しかしながら、"サイズ" 入力コンポーネントも変更することができ、マッピングが生成されるコード、または、MapForce Server により実行される場合、グループのサイズは、マッピングへのパラメーターとして便利に提供することができます。更に詳しい情報に関しては、次を参照してください：マッピングにパラメーターを与える

次に、変数の値はターゲット PersonList XML コンポーネントにより与えられます。作成された各グループのためのファイル名は、**concat** 関数を使用して次の部分を連結することにより計算されます：

1. 各部署の名前
2. 文字列 "_Group"
3. 現在のシーケンス内のグループの番号（例えば、"1" は、この部署の最初のグループを指します）
4. 文字列 "xml"

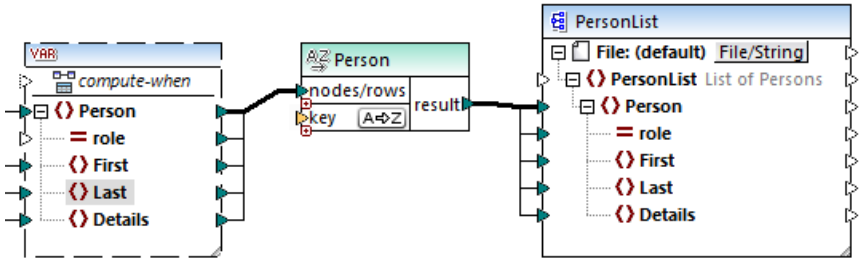
この連結の結果は、変数の Name アイテム内に保管され、ターゲットコンポーネントに動的なファイル名として与えられます。これにより、受信した値のために新規のファイルが作成されます。このサンプルでは、変数は、8個のグループを計算するため、マッピングが実行されると8件の出力ファイルが作成されます。この技術に関する更に詳しい情報は、次を参照してください：複数の入力または出力ファイルを動的に処理.

5.8 データの並べ替え

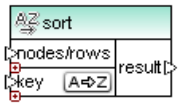
特定のソートキーをベースにした入力データの並べ替えを行う場合、ソートコンポーネントを使用してください。ソートコンポーネントは、XSLT2、XQuery、内蔵の実行エンジンにてサポートされます。変換言語が「Built-in」の場合、並べ替えコンポーネントを使用して、データベーステーブルのデータを並べ替えることができます。しかしながら、パフォーマンスを向上するには、SQL-WHERE/ORDER コンポーネントを使用することにより達成することができます。詳細に関しては、次を参照してください：データベースデータのフィルターと並べ替え（SQL WHERE/ORDER）。

マッピングに並べ替えコンポーネントを追加するには、以下を行います：

- 既存の接続を右クリックして、コンテキストメニューから**「Insert Sort: Nodes/Rows」**を選択します。並べ替えコンポーネントを自動的に挿入し、ソースとターゲットコンポーネントに自動的に接続します。 例えば、下のマッピングでは、並べ替えコンポーネントが変数とXML コンポーネント間に挿入されます。 (並べ替えるフィールドである) 並べ替えキーのみを手動で接続します。



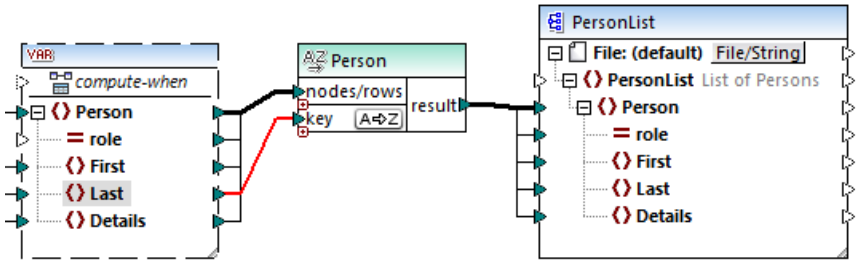
- **「挿入」**メニューから**「並べ替え」**（または、**「並べ替え」**ツールバーボタン）をクリックします。「接続」フォーム内に並べ替えコンポーネントが挿入されます。



ソースコンポーネントに接続が構築されると、タイトルバー名は、`nodes/rows` アイテムに接続されているアイテムの名前に変更されます。

並べ替えるアイテムを定義する：

- 並べ替えコンポーネントの`key` パラメーターを並べ替えるアイテムに接続します。例えば、下のマッピングでは、`Person` nodes/rows は、フィールド`Last` に従い並べ替えられます。

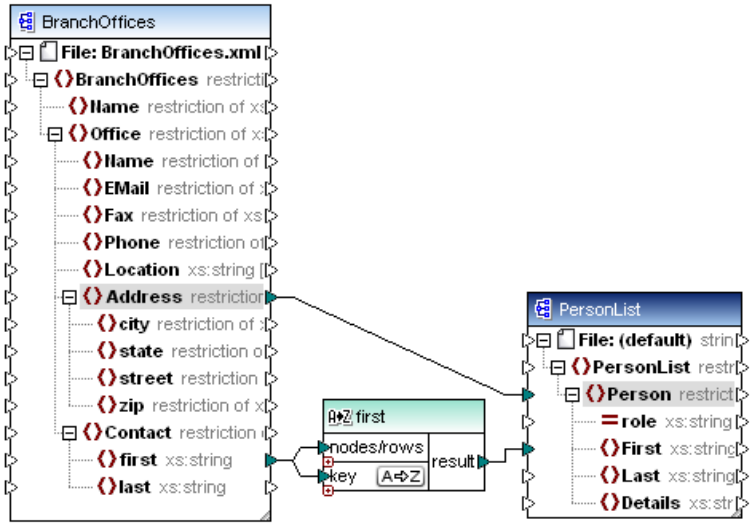


並べ替えの順序の変更：

- 並べ替えコンポーネント内の A⇨Z アイコンをクリックすると、Z⇨A に変更され、並べ替えの順序は逆の順序になったことを示します。

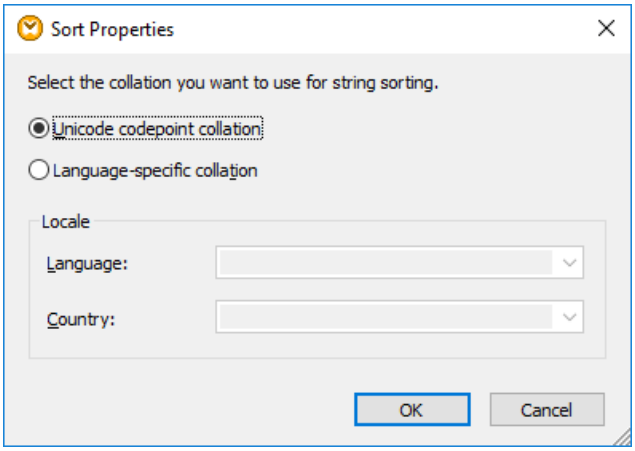
単純型アイテムにより構成される入力データの並べ替え：

- アイテムを並べ替えコンポーネントの `nodes/rows` と `key` パラメーターに接続します。下のマッピングでは、単純型 `first` 要素が並べ替えられます。



言語特有のルールを使用して文字列を並べ替える：

- 並べ替えコンポーネントのヘッダーをダブルクリックして、「並べ替え」プロパティダイアログボックスを開く。



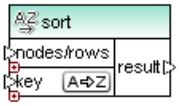
Unicode コードポイント照合：この（デフォルト）オプションにより、コードポイントの値をベースにした比較並べ替えが行われます。コードポイント値とは Unicode コンソーシアムにより採用された Universal Character Set (UCS) における絶対文字に割り当てられた整数のことです。このオプションでは複数の言語やスクリプト間で並べ替えを行うことが可能になります。

言語固有の照合：このオプションにより、並べ替えを行う際にベースとなる言語や国を指定することができます。このオプションは内蔵の実行エンジン（BUILT-IN）が選択されている際にサポートされ、XSLT におけるサポートはコードの実行に使用

されるエンジンに左右されます。

5.8.1 複数のキーを使用した並べ替え

マッピングに並べ替えコンポーネントを追加した後、key という名前の並べ替えキーがデフォルトで作成されます。

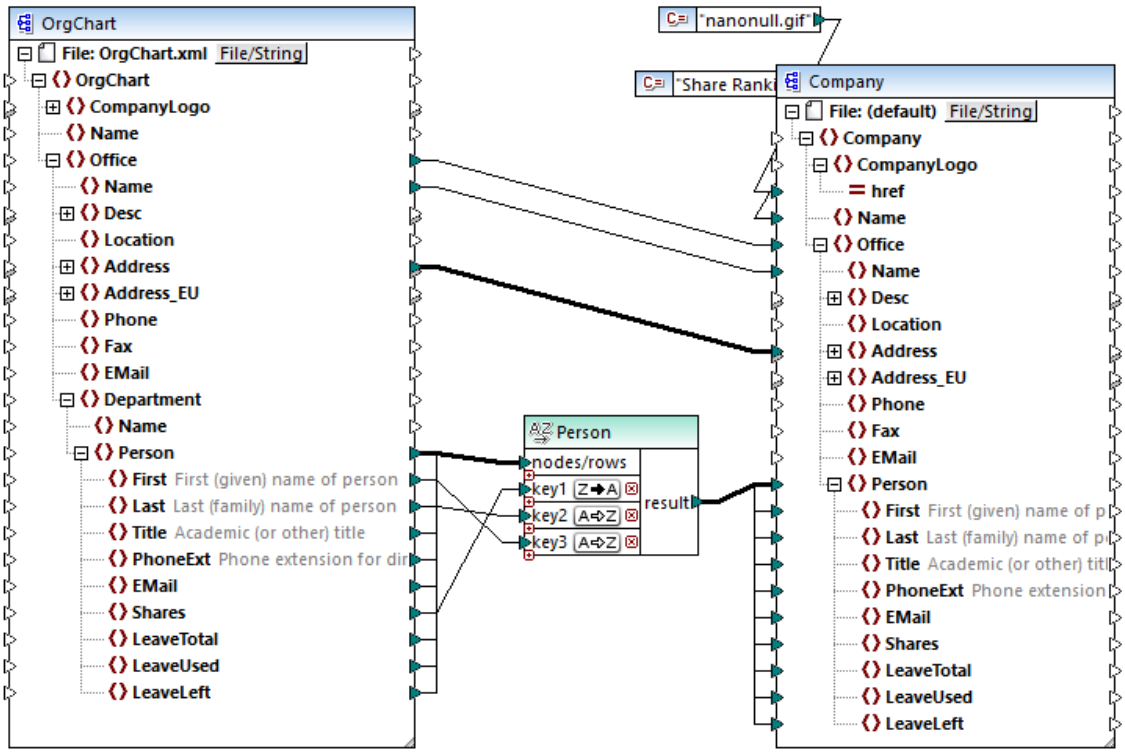


デフォルトの並べ替えコンポーネント

複数のキーを使用して並べ替える場合、並べ替えコンポーネントを以下のように調整します :

- **「キーの追加」**(⊙) アイコンをクリックして新しいキーを追加します (例えば、下のマッピング内の key2).
- **「キーの削除」**(⊗) アイコンをクリックしてキーを削除します。
- ⊙ アイコンに接続をドロップして、キーを追加し、接続します。

複数のキーによる並べ替えを示すマッピングは、次のパスで検索することができます :**<Documents>\Altova \MapForce2017\MapForceExamples\SortByMultipleKeys.mfd**.



SortByMultipleKeys.mfd

上のマッピングでは、Person レコードは3つの並べ替えキーにより並べ替えられています :

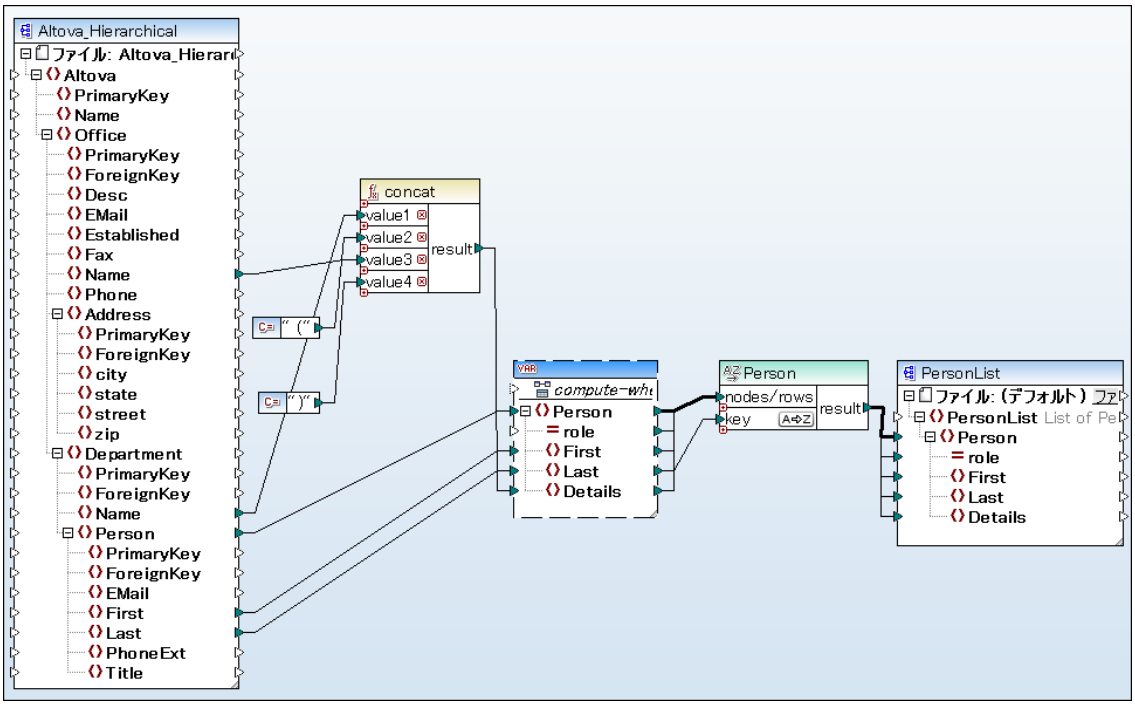
1. Shares (個人が所有するシェア数)
2. Last (姓)
3. First (名)

並べ替えコンポーネント内の並べ替えキーの位置は、並べ替えの優先順序を決定します。例えば、上のマッピングでは、所有するシェア数により並べ替えられています。これは、最も高い優先順位を持つ並べ替えキーです。所有するシェア数が同じの場合も同様で、姓により並べ替えられます。最後に、同数のシェアを持つ同じ性を持つ個人は、名を考慮して並べ替えられます。

各キーの並べ替えは異なる順番であることができます。上のマッピングでは、キー`Shares`は降順の並べ替え順序 (Z-A) が与えられていますが、その他2つのキーには昇順の並べ替え順序 (A-Z) が与えられています。

5.8.2 変数を使用して並べ替える

希望する結果を達成するためにマッピングに中間変数を追加する必要がある場合があります。このサンプルでは、XML ファイルから記録を抽出し、中間変数を使用して並べ替える方法について説明されています。このサンプルには、次のパスで検索することのできるマッピングのサンプルが存在します：**<Documents>\Altova\MapForce2017\MapForceExamples\Altova_Hierarchical_Sort.mfd**。



Altova_Hierarchical_Sort.mfd

このマッピングは、**Altova_Hierarchical.xml** という名前のソースXML ファイルからデータを読み取り、ターゲット XML ファイルに書き込みます。上に示されるとおり、ソースXML には架空の企業の情報が含まれています。企業はオフィスに分岐されています。オフィスは部署に分岐されており、部署は社員に分岐されています。

ターゲットXML コンポーネント、`PersonList`、は`Person` レコードのリストが含まれています。`Details` アイテムは、社員が属するオフィスと部署の情報が保管されています。

ソースXML から全ての社員を抽出し、姓をアルファベット順に並べ替えることが目的です。また、オフィスと部署名は、`Details` アイテムに書き込まれる必要があります。

この目的を達成するために、このサンプルは、次のコンポーネントの型を使用します：

1. **`concat`** 関数：このマッピング内では、この関数は、フォーマット`Office(Department)` で文字列を返しま

す。オフィス名、部署名、かつ開始し終了する2つの定数を入力として取ります。 次も参照してください:関数と作業。

2. 中間変数：変数の役割は同じマッピングコンテキストに個人に関連する全てのデータを運ぶことです。変数によりマッピングがそれぞれの個人のコンテキストで、個人の部署とオフィスを探します。すなわち、変数は、個人が所属するオフィス名と部署名を「記憶」することを意味します。変数が不在の場合、コンテキストは、同じXML スキーマを使用して正確ではなくマッピングは希望しない結果を生成します。 (マッピングの実行方法の詳細に関しては、次を参照してください：マッピングルールと戦略)。変数はXML ファイルの構造を複製することに注意してください。これにより、全てコピー接続を使用して、ターゲットに並べ替えの結果を接続することができます。次も参照してください:変数の使用 と全てコピー接続.
3. 実際の並べ替えを行う、並べ替えコンポーネント：Sort コンポーネントのキー入力は、姓により個人の記録を並べ替える変数のLast アイテムに接続されていることに注意してください。

5.9 フィルターと条件

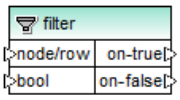
データをフィルターする場合、または値を条件に対して取得する場合、以下のうち1つのコンポーネント型を使用することができます：

- フィルター:Nodes/Rows ( )
- SQL WHERE/ORDER ( )
- If-Else 条件 ( )

挿入 メニューから、または、 **コンポーネントの挿入** ツールバーからこれらのコンポーネントをマッピングに追加することができます。上記のそれぞれのコンポーネントには特定の振る舞いや必要条件があることに注意してください。差異は下のセクションで説明されています。

ノードと行をフィルターする

XML ノードまたはCSV 行を含むデータをフィルターする場合、**Nodes/Rows をフィルター** コンポーネントを使用します。この**Nodes/Rows をフィルター** コンポーネントにより、true または false 条件をベースにし、大きなデータセットからノードのサブセットを抽出することができます。マッピングエリアでのこの構造は以下のように表されます：



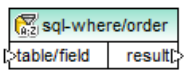
上の構造では、に接続されている条件は、**bool** に接続された**node/row** が、**on-true** または**on-false** 出力に接続されているかを決定します。具体的には、条件がtrue の場合、**node/row** は、**on-true** 出力にダイレクトされます。一方、条件がfalse の場合、**node/row** は、**on-false** 出力にダイレクトされます。

マッピングがフィルターの条件を満たすアイテムのみを消費する必要がある場合、**on-false** 出力を未接続のままにできます。フィルターの条件を満たさないアイテムを処理する必要がある場合、このようなアイテムがダイレクトされる、**on-false** 出力をターゲットに接続します。フィルターの条件が満たされ例外を追加する場合は、**on-false** 出力への接続は必須です（以下を参照してください：例外の追加）。

手順を追ってのマッピングのサンプルは、サンプル：ノードのフィルターを参照してください。.

データベースのデータをフィルター

Nodes/Rows をフィルター コンポーネントは、MapForce により、他の コンポーネント 構造からデータをフィルターすることができます。しかしながら、データベースからデータをフィルターする場合、**SQL WHERE/ORDER** コンポーネントを代わりに 使用することが奨励されます。**SQL WHERE/ORDER** コンポーネントは、データベースと作業し、**Nodes/Rows をフィルター** コンポーネントよりも良いパフォーマンスを与えるために最適化されています。



このようなコンポーネントに関する詳細に関しては、SQL WHERE / ORDER コンポーネントを参照してください。

値を条件付きで返す

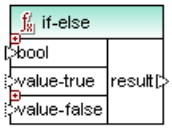
(ノードまたは行ではない)単一の値を条件付きで取得する必要がある場合、**If-Else 条件** を使用します。If-Else 条件は、ノードまたは行をフィルターするには、適していません。**Nodes/Rows をフィルター** コンポーネントとは異なり、**If-**

Else 条件 は、(文字列または整数などの) 単純型を返します。ですから、**If-Else 条件** は、単純型を条件付きで処理するシナリオのみに適しています。例えば、各月の平均気温のリストがあると仮定します :

```
<Temperatures>
   <data temp="19.2" month="2010-06" />
   <data temp="22.3" month="2010-07" />
   <data temp="19.5" month="2010-08" />
   <data temp="14.2" month="2010-09" />
   <data temp="7.8" month="2010-10" />
   <data temp="6.9" month="2010-11" />
   <data temp="-1.0" month="2010-12" />
</Temperatures>
```

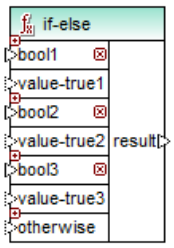
If-Else 条件 により、リスト内の各アイテムのために気温が摂氏 20度を超える場合は、値 "高 "を、気温が摂氏 5度以下の場合は、値 "低 "を返すように設定することができます。

マッピング上で**If-Else 条件** の構造は、以下のようになります :



bool に接続された条件がtrue の場合、**value-true** に接続された**result** としての出力になり、条件がfalse の場合、**value-false** に接続された**result** としての出力になります。**result** のデータ型を事前に知ることはできず、これは値のデータ型、により異なり、**value-true** または**value-false**. に接続された値により異なります。重要な点は、常に (文字列および整数などの) 単純型であるということです。入力値を (ノードまたは行などの) 複合型に接続することは**If-Else 条件**によりサポートされていません。

If-Else 条件は拡張することが可能です。これは、**追加** ( ) ボタンをクリックすることにより、コンポーネントに複数の条件を追加できることを意味します。以前に追加された条件を削除するには、ボタンを**削除** ( ) をクリックします。この機能により、条件がtrue の場合は、複数の条件をチェックして、各条件のために異なる値を返すことができます。



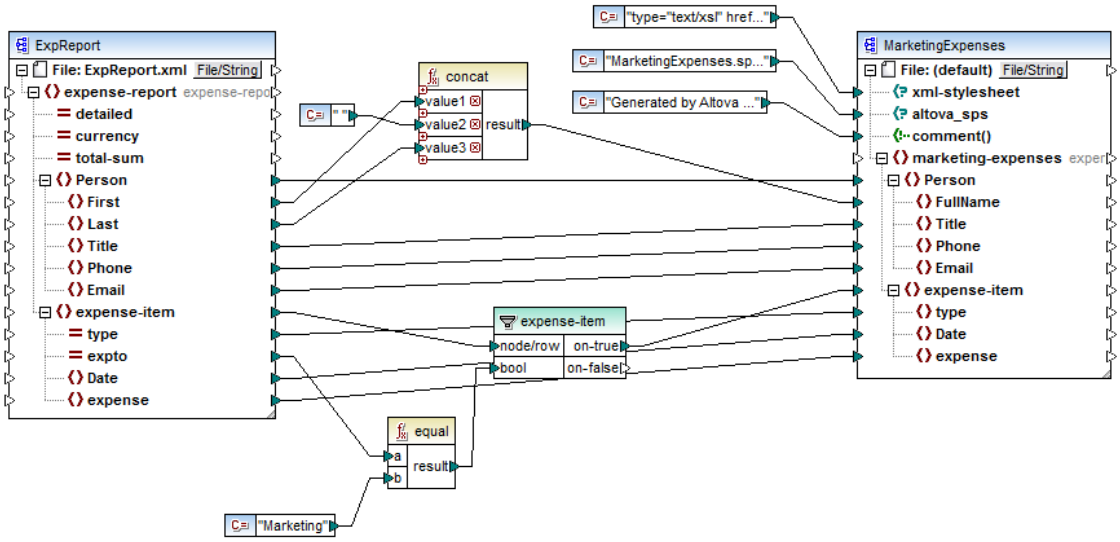
展開された**If-Else 条件** は上から下へと評価されます (最初の条件が最初にチェックされ、2番目が次にチェックされます)。条件の全てが真ではない場合に値を返す場合は、**otherwise** に接続してください。

手順を追ってのマッピングのサンプルは、例 :値を条件付きで返すを参照してください。

5.9.1 サンプル :ノードのフィルター

この例は、true/false 条件を基にしてノードをフィルターする方法について説明しています。**Filter: Nodes/Rows** ( ) コンポーネントがこの目的を達成するために使用されています。この例で説明されている技術は、XML のみではなく、CSV またはテキストなどの他のコンポーネント型などとでも作業することができます。データベースの場合、フィルターを使用することはできますが、パフォーマンスのためには、代わりにSQL WHERE/ORDER コンポーネントを使用することが奨励されます (以下を参照してください:SQL WHERE / ORDER コンポーネント)。

この例で説明されているマッピングは以下のパスで検索することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\MarketingExpenses.mfd**。



上に示されるように、このマッピングは、経費の方向ｋ ("ExpReport") のデータを含むソースXML からデータを読み取り、ターゲットXML ("MarketingExpenses") に書き込みます。ターゲットとソースの間には複数の他のコンポーネントが存在します。最も関連性の高いコンポーネントは、このトピックの主題を表す **expense-item** フィルター（ ）です。

このマッピングのゴールは、マーケティング部署に属する経費アイテムのみをフィルターすることです。この目的を達成するために、フィルター コンポーネントがマッピングに追加されました。（フィルターを追加するには、**「挿入」**メニューをクリックして、**「Filter: Nodes/Rows」**をクリックします）。

各経費がマーケティング部署に属するかを識別するために、このマッピングは、ソース内の "expto" 属性の値を確認します。この属性が経費がマーケティングの経費である場合、値 "Marketing" を有します。例えば、下にリストされるコードでは、最初と3番目のアイテムは、マーケティングに属し、2番目のアイテムは開発部署に、4番目は販売部署に属することが示されています：

```
...
   <expense-item type="Meal" expto="Marketing">
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Development">
      <Date>2003-01-02</Date>
      <expense>122.12</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Marketing">
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <expense-item type="Entertainment" expto="Sales">
      <Date>2003-01-02</Date>
      <expense>13.22</expense>
   </expense-item>
...
```

マッピングが実行される前のXML 入力

マッピングエリアで、フィルターの**node/row** 入力が、ソースコンポーネント内の**expense-item** ノードに接続されていま

す。これにより、フィルターコンポーネントが処理されなければならないノードのリストを取得することを保証します。

フィルタリングが発生する条件を追加するには、MapForce core ライブラリから**equal** 関数 を追加しました (詳細に関しては、以下を参照してください:関数と作業する)。**equal** 関数は、"type" 属性を値 "Marketing" を持つ定数と比較します (定数を追加するには、**挿入**」メニュー をクリックして、**定数**をクリックします)。

条件を満たすアイテムのみをフィルターする必要があるため、フィルターの**on-true** 出力のみをターゲットコンポーネントに接続します。

マッピングの結果をプレビューする準備が整うと、**出力**」タブをクリックします。MapForce は、フィルターの**bool** 入力に接続されている条件である、それぞれの経費アイテムノードを評価します。条件がtrue の場合、経費アイテムノードはターゲットにパスされます。それ以外の場合は、無視されます。結果、条件を満たす経費アイテムのみが出力内に表示されます:

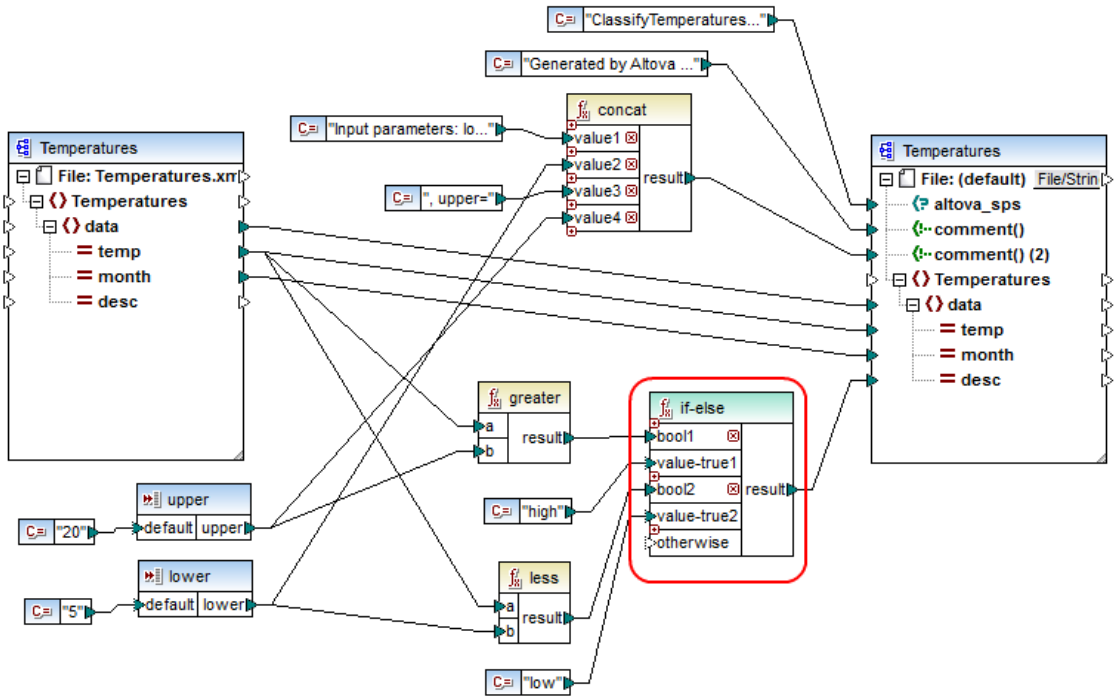
```
...
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
...
```

マッピングの実行後のXML 出力

5.9.2 サンプル :条件付で値を返す

この例では、true/false 条件を基にして、コンポーネントから単純型の値を返す方法を説明しています。**If-Else 条件** ( ) がこの目的を達成するために使用されています。 **If-Else 条件** とフィルターコンポーネントを混同しないようご注意してください。**If-Else 条件** は、(文字列、整数などの) 単純な値を条件付きで処理する場合のみに適しています。ノードなどの複合値をフィルターする場合、代わりにフィルターを使用してください (以下を参照してください:例 : ノードのフィルター)。

この例で説明されるマッピングは以下のパスで検索することができます :**<Documents>\Altova\MapForce2017\MapForceExamples\ClassifyTemperatures.mfd**。



このマッピングは、気温データを含むソースXML ("Temperatures") からデータを読み取り、データを同じスキーマに準ずるターゲットXML に書き込みます。ターゲットとソースの間には複数の他のコンポーネントが存在します。そのうちの1つが、このトピックの主題である (赤でハイライトされている) **if-else** 条件です。

このマッピングのゴールは、ターゲット内の各気温の記録に短い説明を追加することです。具体的には、摂氏 20度以上の気温の説明は、 "high" と表示され、摂氏 5度以下の気温は、 "low" と表示されるように設定します。上記以外の場合は、説明が表示されません。

この目的を達成するために、条件付き処理が必要とされます。ですから、If-Else 条件 がマッピングに追加されました。(If-Else 条件を追加するには、 **挿入** メニューをクリックして、**If-Else 条件**をクリックします)。このマッピングでは、以下の2つの条件を受け入れるために (▫ ボタンを使用して) If-Else 条件が展開されています :**bool1** と**bool2**。

条件自身は、MapForce core ライブラリから追加された**greater** と**less** 関数により追加されます (詳細に関しては、以下を参照してください:関数と作業する)。 関数は、 "upper" と "lower" と呼ばれる2つの入力コンポーネントより与えられた値を評価します (入力コンポーネントを追加するには、 **挿入** メニューをクリックして、 **入力の挿入** をクリックします。入力コンポーネントに関する詳細については、 マッピングにパラメーターを提供するを参照してください

greater と**less** 関数は、true または false のどちらかを返します。関数の結果は、ターゲットインスタンスに何か書き込まれるかを決定します。具体的には、ソース内の "temp" 属性の値が、20 よりも大きい場合、定数値 "high" が **if-else** 条件にパスされます。ソース内の "temp" 属性の値が 5 よりも小さい場合、定数値 "low" は、**if-else** 条件にパスされます。**otherwise** 入力が接続されていませんので、上記の条件が満たされない場合は、**result** 出力コネクターにパスされる値はありません。

最後に、**result** 出力コネクタは、(各気温の記録に1度ずつ与えられる) この値をターゲット内の "desc" 属性に与えます。
マッピングの結果をプレビューする準備が整うと、 **出力** タブをクリックします。気温が20度より高い、または、5度より低い場合、結果 XML 出力は、 "desc" 属性を含むことに注意してください。

```
...
   <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"/>
   <data temp="16.2" month="2006-05"/>
   <data temp="19" month="2006-06"/>
   <data temp="22.7" month="2006-07" desc="high"/>
   <data temp="23.2" month="2006-08" desc="high"/>
...
```

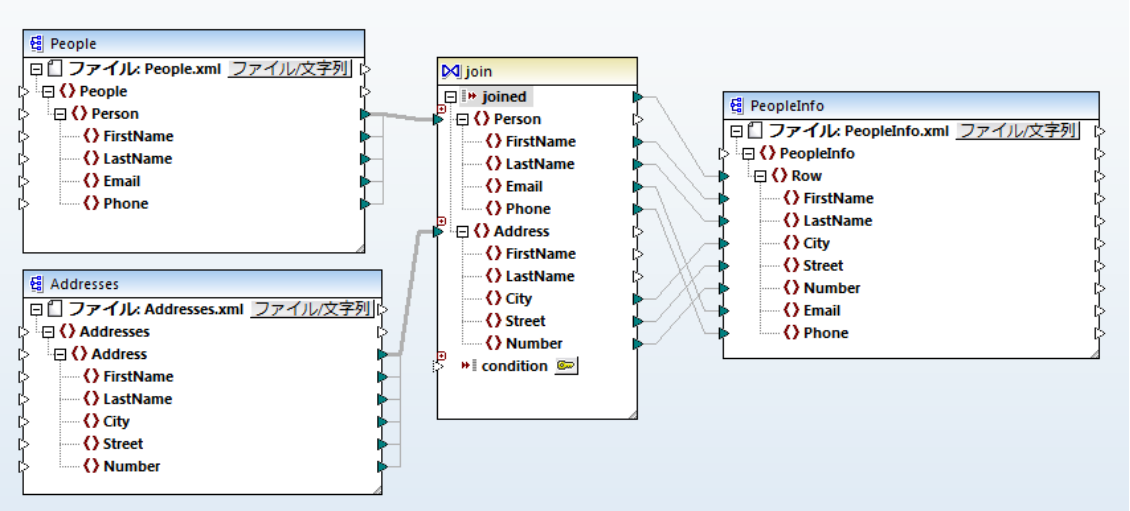
マッピングの実行後のXML 出力

5.10 データのジョイン

(例えば、最初の構造内の フィールドA が 2番目の構造内のフィールドBと同じ値を持つなど)の条件をベースに2つの構造からのデータを結合する必要がある場合があります。このようなマッピングの条件には、ジョインコンポーネントを使用することができます。

ジョインコンポーネントは、カスタムに定義付けられた条件をベースにして2つの構造をジョインすることを有効化する MapForce コンポーネントです。条件を満たすアイテムのアソシエーション (ジョインされたセット)を返します。ジョインは、(ID などの)共通のフィールドを共有する2つの構造からのデータを結合する際にとても役に立ちます。

例えば、下のマッピングでは、中間コンポーネントは 『ジョイン』 コンポーネントです。このマッピングでは、 2つのXML 構造 (個人のリストと住所のリスト)がジョインされます。このサンプルの目的は、各個人の情報をターゲットXML ファイルに書き込むことです。 FirstName とLastName フィールドは、ジョインするキーとしての役割を果たします。 すなわち、(Person の下の)FirstName とLastName の値が(Address の下の) FirstName とLastName と同じ値の場合、同じ個人に属する住所の詳細は 『ジョイン』されます。ジョインした構造からのアイテムは、次のターゲット (この場合、 XML ファイル)にマップすることができます。ジョインの条件は、ジョインコンポーネント プロパティ内で **『ジョインの条件の定義』** ( ) ボタンをクリックして定義されます。このサンプルにはマッピングのサンプルが存在し、サンプル: XML 構造のジョイン内で詳しく説明されています。



JoinPeopleInfo.mfd

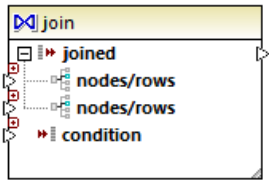
上記のとおり、ソース構造 とジョインコンポーネントは、マッピングを優先順位の低い 『全てコピー』 接続を使用して接続されています。一般的には、コンテキストが関連する場合、このような接続は自動的にMapForce により作成されます (詳細に関しては、次を参照してください:全てコピー接続)。

(上のマッピングのような)個別のコンポーネント、または、同じコンポーネントに属するコンポーネントから2つの構造をジョインすることができます。(例えば、XML 構造とデータベーステーブルなどの)異なる型 の構造をジョインすることもできます。データベースに関連したジョインの詳細に関しては、次を参照してください:データベースのデータのジョイン (SQL JOIN)。

ジョインコンポーネントを追加する

1. マッピング変換言語をBUILT-IN に設定します (これを行うには BUILT-IN ツールバーボタンをクリックします、または、 **出力 | Built-In 実行エンジン** 』メニューコマンドを使用します)。
2. **『挿入』** メニューから **『ジョイン』** をクリックします。または、 **『ジョイン』** ( ) ツールバーボタンをクリックします。ジョインコンポーネントがマッピングに表示されます。デフォルトでは2つの構造からデータを受け入れるため、2つの

nodes/rows 入力が存在します。必要であれば、**入力の追加**(⊡)ボタンをクリックして、ジョインに新しい入力を追加することができます。以下を参照してください[3つ以上の構造をジョインする](#)。



3. ジョインコンポーネントのnodes/rows アイテムとジョインされる構造を選択します。
4. ジョイン(または複数の条件)のための条件を追加してください。これを行うには、ジョインコンポーネントを右クリックし、**プロパティ**を選択します。ジョインの条件は、関数のブール値の結果をジョインコンポーネントのcondition アイテムに接続することにより、マッピングから直接追加することができます。データベースがジョインされる特定の場合、ジョインの条件は自動的にMapForce により作成されます。詳細に関しては、次を参照してください:[ジョインの条件の追加](#)。

メモ:

- ジョインコンポーネントは、BUILT-IN 実行エンジン内でのみサポートされています。C#、C++、またはJava 内でのコードの生成はサポートされていません。
- 構造がサポートされていない、または、ジョインのための入力ソースによりサポートされて以内場合、MapForce マッピング上直接ヒントを表示するか、または、マッピングを検証する際に、メッセージウィンドウ内にヒントを表示します。マッピング(次を参照してください:[マッピングの検証](#))。
- ジョインコンポーネントをインラインユーザー定義関数の結果または入力パラメーターに接続しないでください。このような接続が存在すると、マッピングの検証中に検証エラーが発生します。
- 使用することのできるデータベースコンポーネント(テーブルまたはビューなどの)を直接ジョインコンポーネントに接続すると、**SQL モード**(SQL)ボタンがジョインコンポーネントの右上に自動的に表示されます。有効化されると、このボタンは、ジョイン操作に適用することのできる特別なSQL 機能を提供します(次を参照してください:[SQL モード内のジョインについて](#))。
- 3つ以上の構造をジョインする必要がある場合、複数のジョインコンポーネントを追加して、「重ねて」表示します(すなわち、ジョインコンポーネントの出力を他のジョインコンポーネントの入力として与えます)。ジョインされたシーケンス全部を他のジョインの入力としては使用できないことに注意してください。ジョインされたアイテムの出力を他のジョインコンポーネントに接続しようと試みると、エラーメッセージが表示されます。しかしながら、ジョインの結果の一部を他のジョインに接続することは可能です。
- SQL モードでは、ジョインを重ねて表示することはサポートされていません。しかしながら、ジョインされた子孫(子)テーブルを他のジョインに接続することは可能です。代わりに、SQL SELECT ステートメントを使用することができます(次を参照:[仮想テーブルとしてのSQL SELECT ステートメント](#))。親と子テーブルに関する詳細は、次を参照してください:[データベースリレーションシップの扱い方](#)。

他のコンポーネント型と比較したジョインコンポーネント

一部の場合、同じ結果を得るために、ジョインコンポーネントの代わりに複合型変数、または、フィルターを使用することができます。次を参照してください:[変数の使用](#) と[フィルターと条件](#))。しかしながら、他のコンポーネント型とは異なり、ジョインコンポーネントは、同じコンテキストに与えられるデータを確認することができるため、マッピングのロジックをわかり易くします。また、SQL モードがジョインコンポーネント上で有効化されている場合、マッピングのパフォーマンスが大幅に向上します(これはデータベースジョインにも適用されます、次を参照してください:[データベーステーブルのジョイン](#))。

親コンテキストの追加

特別な場合は、特定のマッピングの結果を達成するために、ジョインコンポーネントに接続されているデータのために「いわゆる「親コンテキスト」と呼ばれる)マッピングのコンテキストを明示的に提供することができます。親コンテキストを追加するには、ジョインコンポーネントのジョインされたアイテムをクリックし、コンテキストメニューから**親コンテキストの追加**」を選択します。

ジョインコンポーネントは、必要とするソースアイテムに接続することのできる追加 parent-context 入力を含む外観に変更します。詳細に関しては、以下を参照してください マッピングコンテキストのオーバーライド。

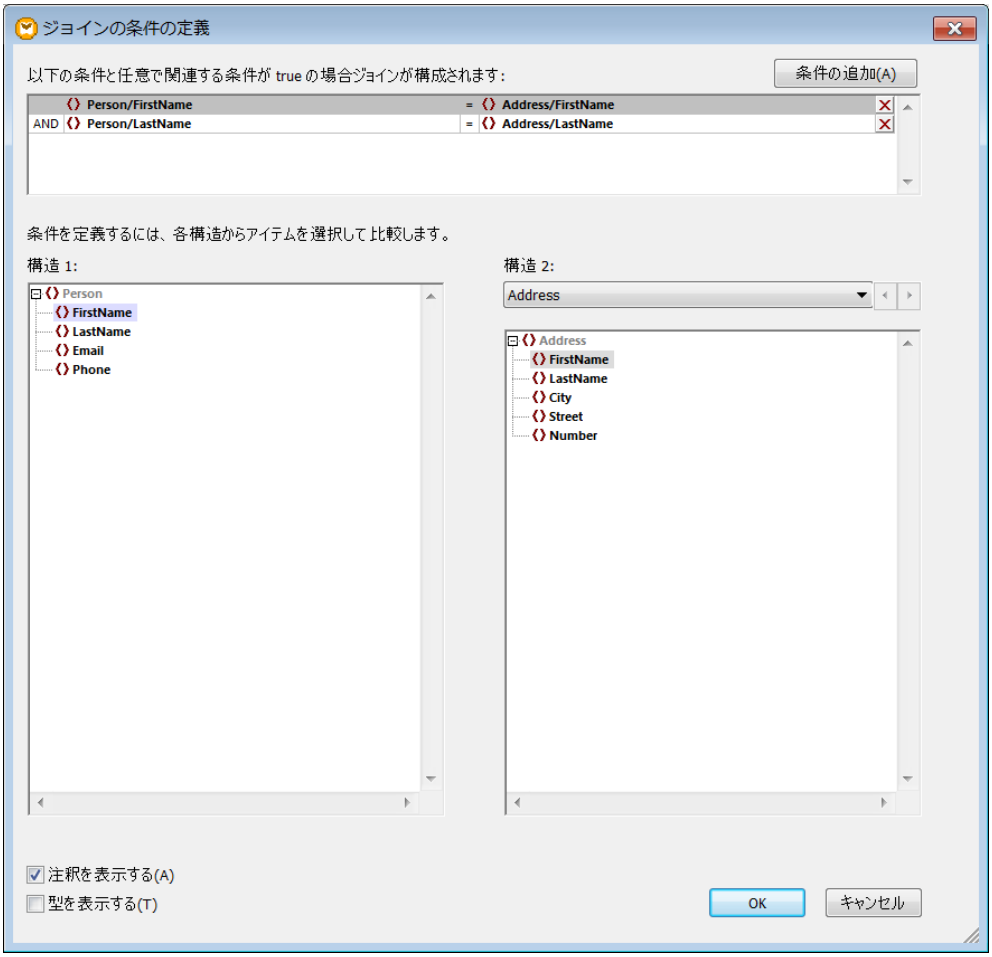
5.10.1 ジョインの条件の追加

条件に従い2つの構造のアイテムを組み合わせることによりジョインは作動します。このため、ジョインには常に条件が必要になります。下に示されるようにジョインの条件を追加するには複数の方法があります。

メモ: SQL モード内でデータベーステーブルがジョインされると、MapForce は、テーブル間で検出された外部キーリレーションシップをベースにして、ジョインの条件を自動的に作成します。自動的なジョインの条件が起こるには、データベーステーブルが MapForce コンポーネント上で親子関係内に存在する必要があります（すなわち、1つのテーブルが「親」または「子」で、他のテーブルがコンポーネント上に存在する必要ああります）。サンプル：SQL モード内のテーブルのジョインを参照してください。

アプローチ 1：コンポーネント プロパティからジョインの条件を追加する

1. マッピング上で、2つ以上の構造（または、データベーステーブル）がジョインコンポーネントに接続されている必要があります。このサンプルで説明されているジョインコンポーネントは、フォルダー **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** 内で使用することのできる **JoinPeopleInfo.mfd** マッピングの一部です。詳細に関しては、サンプル：XML 構造のジョインを参照してください。
2. ジョインコンポーネントで、**「ジョインの条件の定義」**（ ）ボタンをクリックします（または、コンポーネントのヘッダーを右クリックし、コンテキストメニューから **プロパティ** を選択します）。
3. 左の構造からアイテムを選択し、その他を右の構造から選択します（これは、このペアの比較が true を返すと、左と右の構造がジョインされます）。



複数の条件を追加する必要がある場合、をクリックして、 **条件の追加**」新しいアイテムのペアを選択します。例えば、上のイメージは内では、2つのジョインの条件が定義されています:

1. 構造 1内の`FirstName` は構造 2内の`FirstName` と等価である必要があります。
2. 構造 1内の`LastName` は、構造 2内の`LastName` と等価である必要があります。

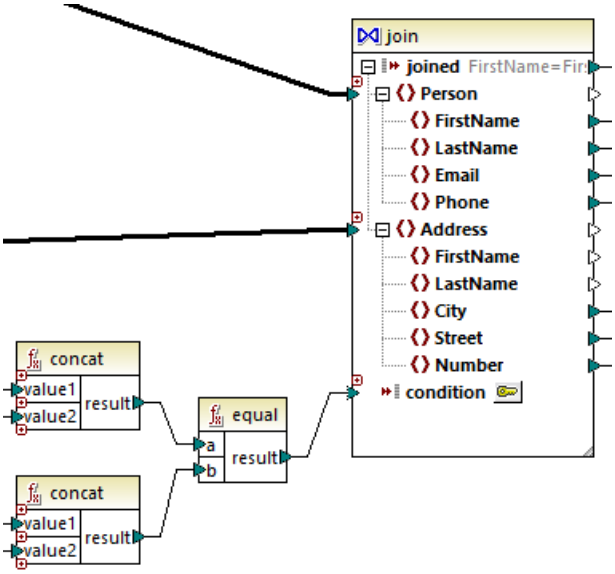
ジョインの条件を削除するには、横の **削除**」 ボタンをクリックします。

メモ: 複数のジョインの条件が存在する場合、 2つの構造がジョインされるためには、全ての条件が満たされている必要があります。すなわち、複数の条件は理論的なAND オペレーションによりジョインされていることを意味します。また、マッピングから追加された任意の条件も含んでいます (次を参照してください:下のアプローチ 2)。

- 2つ以上の構造がジョインコンポーネントに接続されている場合、これらの追加構造は、 構造 2」の下のドロップダウンリストに表示されます。このような追加構造をドロップダウンリストから選択すると、左側のペインに、ジョインコンポーネント前の全ての構造が表示されます。この方法で、任意の複数の構造間のジョインの条件を定義することができます。例に関しては、以下を参照してください サンプル:CSV レポートを複数のテーブルから作成する。
- 各構造内のアイテムのデータ型をビューするためには、 **型の表示**」チェックボックスを選択します。基になるスキーマ (またはデータベース)内に情報が存在することが前提で、 **注釈の表示**」オプションは、アイテムの追加情報を表示します。両方のチェックボックスが選択されていると、レイアウトは、注釈と型の両方を表示するために調整されます。例:

アプローチ 2：マッピングからジョインの条件を追加する

- マッピング上で、ブール値を生成するコンポーネントを追加し、condition アイテムの入力にブールの出力を接続します。例えば、**equal** 関数は、一部のマッピングアイテムの値を比較し、ジョインコンポーネントの condition アイテムの入力として、ブール値の結果を提供します。



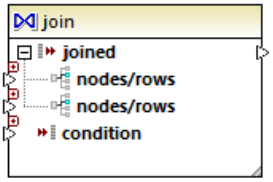
メモ： ジョインコンポーネント プロパティ (アプローチ 1) で条件が定義されていない場合、ジョインコンポーネントの condition アイテムが接続されている必要があります (アプローチ 2)。

アプローチ 3：複合型のアプローチ

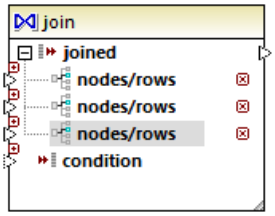
同じマッピング内で、コンポーネント プロパティ (アプローチ 1) 内のジョインの条件を定義し、マッピング (アプローチ 2) からの1つと結合することができます。しかしながら、SQL モードを使用して、データベーステーブルをジョインする場合は、条件はアプローチ 1 を使用して定義される必要があります (次を参照してください：SQL モード内のジョインについて)。

5.10.2　3つ以上の構造をジョインする

メニューコマンド **挿入 | ジョイン** を使用して、ジョイン コンポーネントをマッピングに追加する場合、2つの構造をデフォルトで受け入れることができます (すなわち、コンポーネントには 2つの nodes/rows 入力が含まれています)。



2つの構造をジョインする場合、**入力を追加** (⊞) ボタンをクリックし、必要な数だけ nodes/rows を作成します。nodes/rows 入力を削除する場合、**入力を削除** (⊠) ボタンをクリックします。ジョインは少なくとも 2つの構造を必要とします、ですから ⊠ ボタンは、2つの入力が存在する場合のみ使用することができます。



ジョインに複数の入力が存在する場合、ジョインの条件は、ジョインする各入力を考慮する必要があります。以下を参照してください ジョインの条件を追加する。複数のデータベーステーブルをジョインするサンプルのステップバイステップの例に関しては、以下を参照してください サンプル：CSV レポートを複数のテーブルから作成する

5.10.3 サンプル：XML 構造のジョイン

このサンプルは、ジョインコンポーネントを使用して、2つの XML 構造からのデータを結合する方法について説明しています。このサンプルは、次のパスにあるマッピングサンプルが存在します：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\JoinPeopleInfo.mfd**.

このマッピングの目的は、2つの ソース XML ファイルから個人の情報（名、姓、住所、電子メール、と電話番号）を収集し、単一のターゲット XML ファイルに書き込むことです。

最初の XML ファイルは、各個人の名と姓、電子メールと電話番号を、下のサンプルコードで示されているとおり保管します（XML 宣言、名前空間、とレコードの一部は簡素化のために省略されています）：

```
<People>
   <Person>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Person>
   <Person>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Person>
</People>
```

People.xml

2 番目の XML ファイルは、各個人の名と姓、および電子メールアドレスの詳細を保管します。：

```
<Addresses>
   <Address>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
   </Address>
   <Address>
      <FirstName>Totie</FirstName>
```

```
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
   </Address>
</Addresses>
```

Addresses.xml

マッピングの目的は、<Address> 情報を持つ最初のファイルからの<Person> 情報と２番目のファイルからの情報を姓が一致するか比較します。特に、 最初のファイル内の各 <Person> と ２番目のファイル内の各 <Address> で、FirstName とLastName が比較されます。両方の値が同じの場合、対応する<Person> と<Address> レコードが同じ個人を参照している場合、ジョインが実行されます。 ターゲットXML 構造 は以下のようになります：

```
<PeopleInfo>
   <Row>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Row>
   <Row>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Row>
</PeopleInfo>
```

PeopleInfo.xml

このマッピングの目的は、ジョインコンポーネントをマッピングに追加することで簡単に達成することができます。他のコンポーネント型を使用して、同じ結果を達成することもできることに注意してください。しかしながら、下のステップでは、このサンプルの主題である、ジョインコンポーネントを使用してサンプルが説明されています。

必要とされるマッピングを作成するには、以下のステップに従います。

ステップ 1：ソースXML ファイル をマッピングに追加する

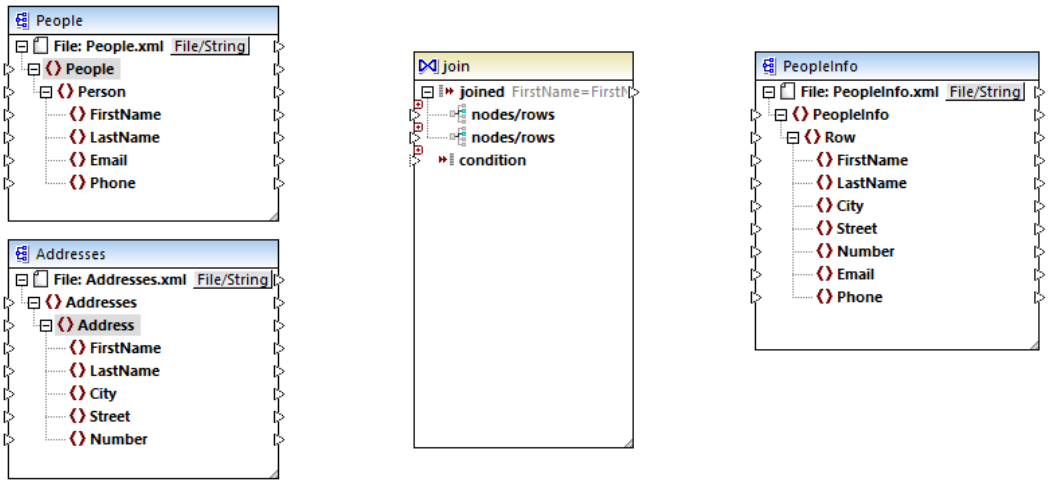
1. **挿入**」メニューからXML **スキーマ/ファイル** をクリックします。次の ソースファイルを参照します：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\People.xml**。
2. **Addresses.xml** (2 番目のソースファイル)のために上のステップを繰り返します。

ステップ 2：ターゲット スキーマファイルをマッピングに追加する

- **挿入**」メニューからXML **スキーマ/ファイル** をクリックします。 **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\PeopleInfo.xsd** (ターゲットXSD スキーマファイル)を参照しますXML ファイルを与えるようにプロンプトされると、 **【スキップ】** をクリックします。ルート要素を選択するようにプロンプトされると、ルート要素としてPeopleInfo を選択します。

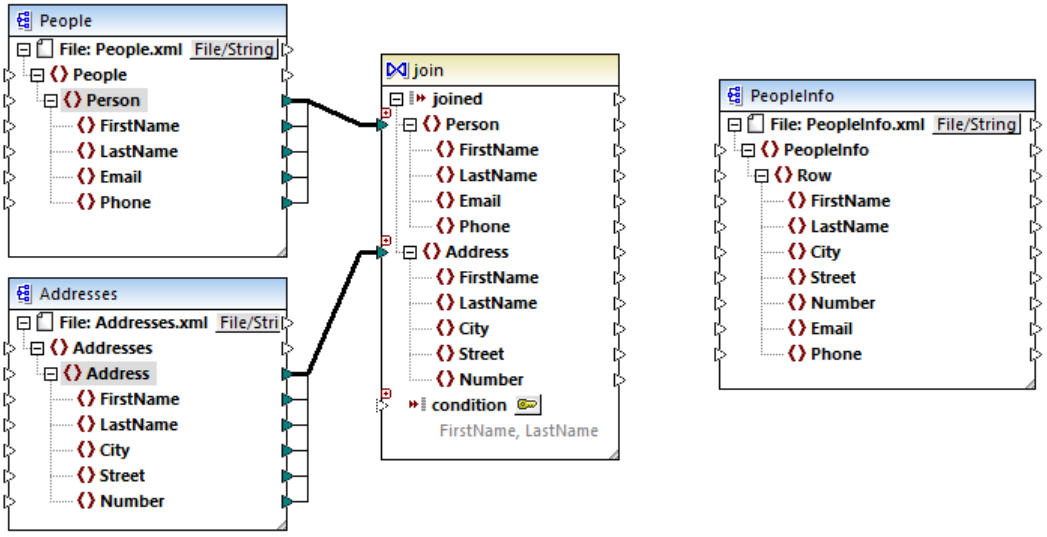
ステップ 3：ジョインコンポーネントを追加する

1. **「挿入」**メニューから**「ジョイン」**（または、**「ジョイン」**ツールバーボタン）をクリックします。この時点でマッピングは以下のようになります（下のマッピングと同じ表示にするには、コンポーネントをドラッグしてサイズを調整する必要があります）：

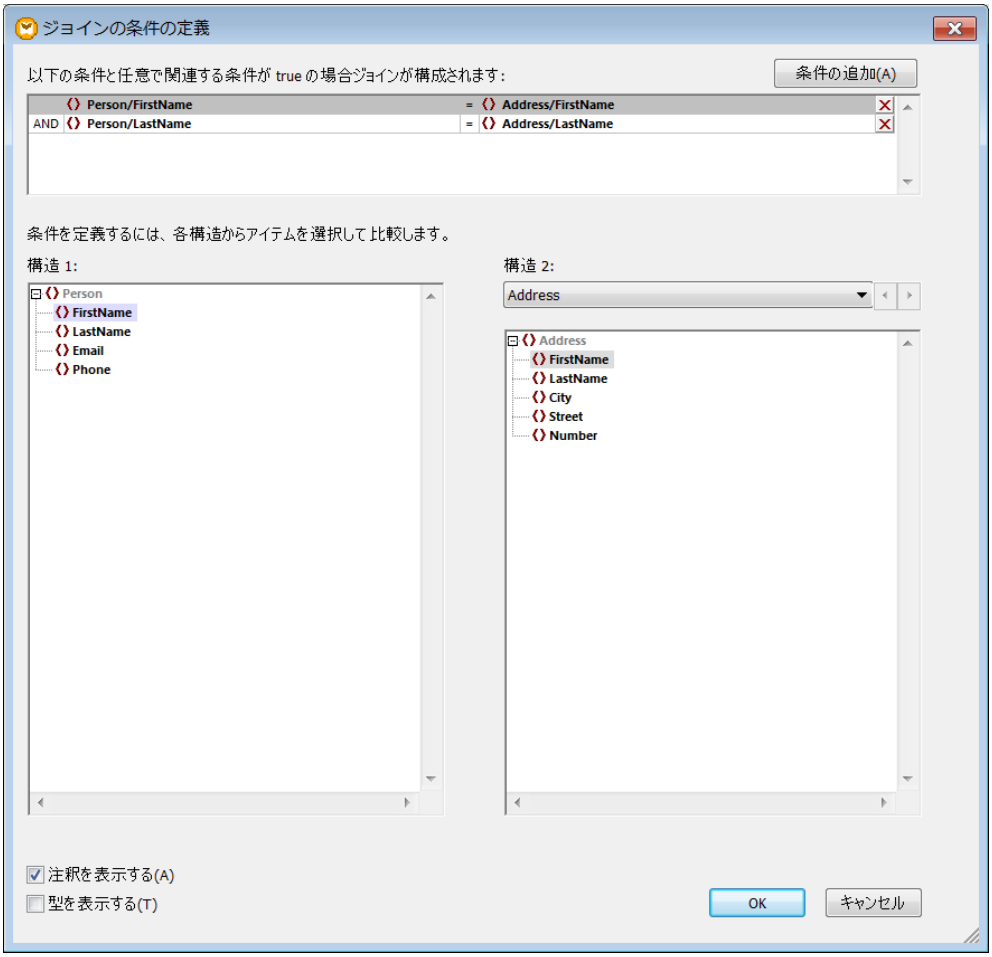


ジョインコンポーネントの構造に注目してください。比較する2つの構造（この場合、`Person` と`Address` 構造）を接続する 2つのnodes/rows アイテムが存在します。

2. ジョインコンポーネントの最初の`nodes/rows` アイテムに`Person` から接続を描きます。同様に、2番目の`nodes/rows` アイテムに`Address` を接続します。



3. 既に説明されているとおり、`FirstName` と`LastName` 値が両方の構造内で等価の場合のみジョインは実行されます。この条件を定義するために、**「ジョインの条件の定義」**ボタンをクリックしてください。
4. 最初のジョインの条件（構造 1の下の`FirstName` と構造 2の下の`FirstName`）を定義するアイテムのペアを選択します。
5. **「条件の追加」**をクリックして、上記のステップを`LastName` のために繰り返します。



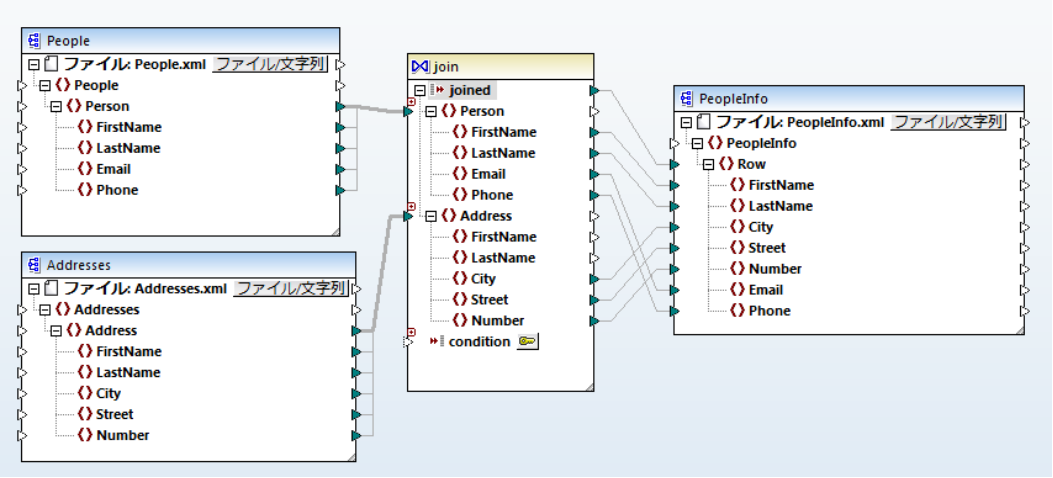
マッピングの一部では、1つの比較から構成される条件のみでジョインを実行することを可能な場合があります。しかしながら、このサンプルでは、2つの比較が作成されることが重要です:

1) 構造1内の`FirstName` = 構造 2内の`FirstName`。
2) 構造1内の`LastName` = 構造 2内の`LastName`。

複数の条件が定義されている場合、ジョインが実行されるには全ての条件が満たされている必要があります。このため、このサンプルでは、(期待される結果である)ジョインは両方の比較の条件が満たされている場合のみ実行されます。それ以外の場合、1つの比較の条件のみが定義されている場合、ジョインは同じ名と姓を持つ個人に対して実行される可能性があります。

ステップ 4:ジョインコンポーネントをターゲット スキーマにマップする

2つの構造はジョインされ、ジョインした構造のどのアイテムがターゲットにマップされるかを定義することができます。これを行うには、ターゲットコンポーネントへジョインされた構造のアイテムからの接続を下に示されるように作成します。`joined` と`Row` 間の接続の目的は次のとおりです:ジョインの条件が満たされると、ターゲット内に新規の`Row` アイテムを作成する。



出力 タブをクリックしてマッピング出力をプレビューすることができます。期待通り、各レコード (<Row>) には、2つの異なるソースからジョインされた住所の詳細が含まれています。

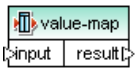
5.11 Value- Map の使用

Value-Map コンポーネントを使用することで、与えられた入力値から、ルックアップテーブルを使った別の値へ出力値を変換することができます。この機能は列挙型の変換を行う際に便利です。コンポーネントには 1 つの入力と 1 つの出力アイテムだけが含まれます。

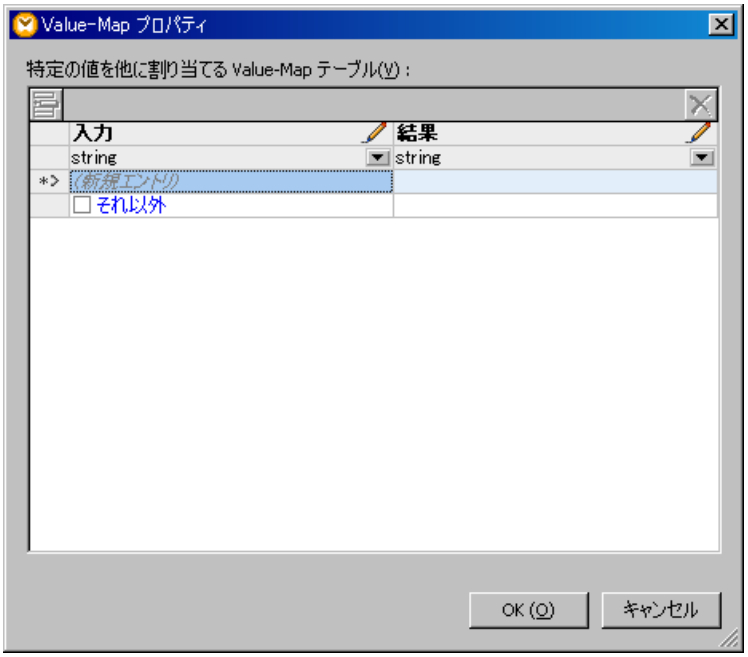
メモ:特定の条件に従ってデータの取得 (フィルタリング)を行いたい場合は、**フィルター**コンポーネントを使用してください。フィルターと条件のセクションを参照してください。

Value- Map コンポーネントを使用する:

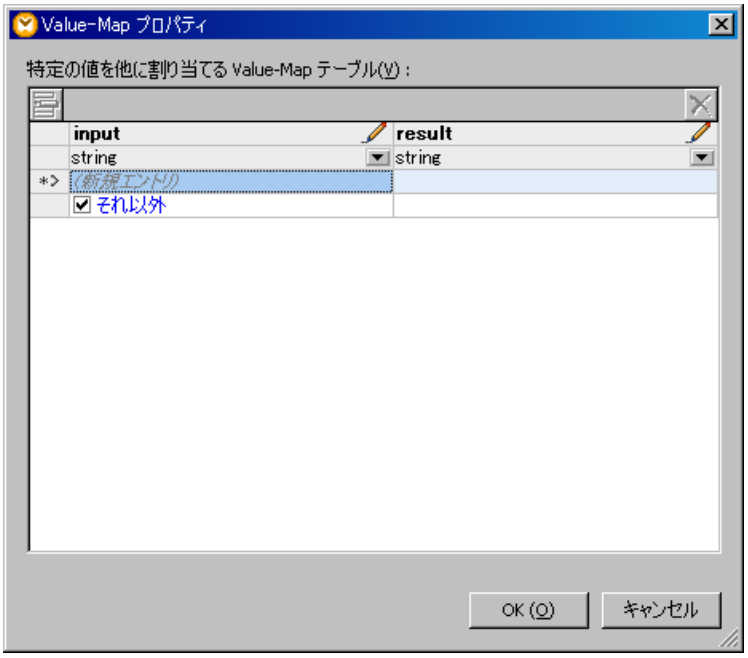
1. メニューオプションの「挿入 | Value-Map」を選択するか、アイコンバーにある Value-Map アイコン  をクリックします。



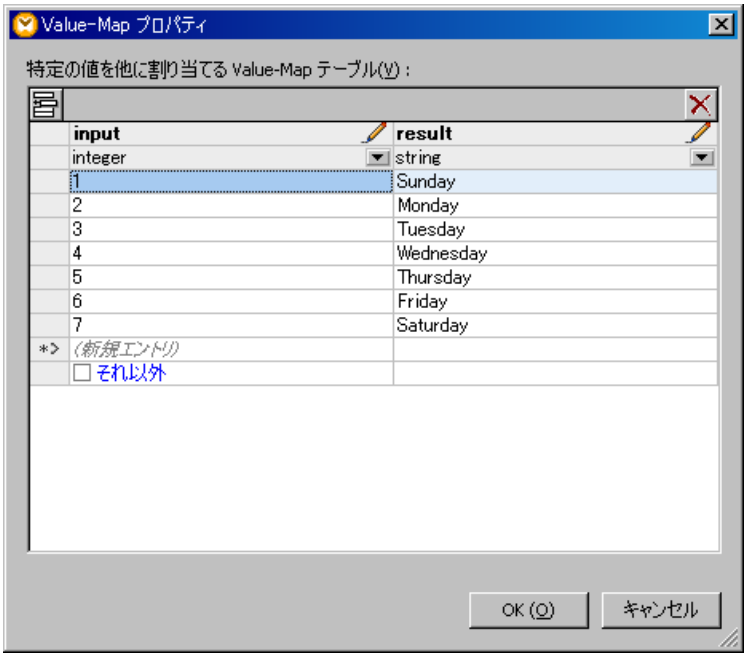
2. Value-Map コンポーネントをダブルクリックして、Value-Map テーブルを開きます。



3. C カラムヘッダーをクリックして、最初のカラムに **Weekday input** と、そして 2 つめのカラムに **Day of the Week** と入力します。



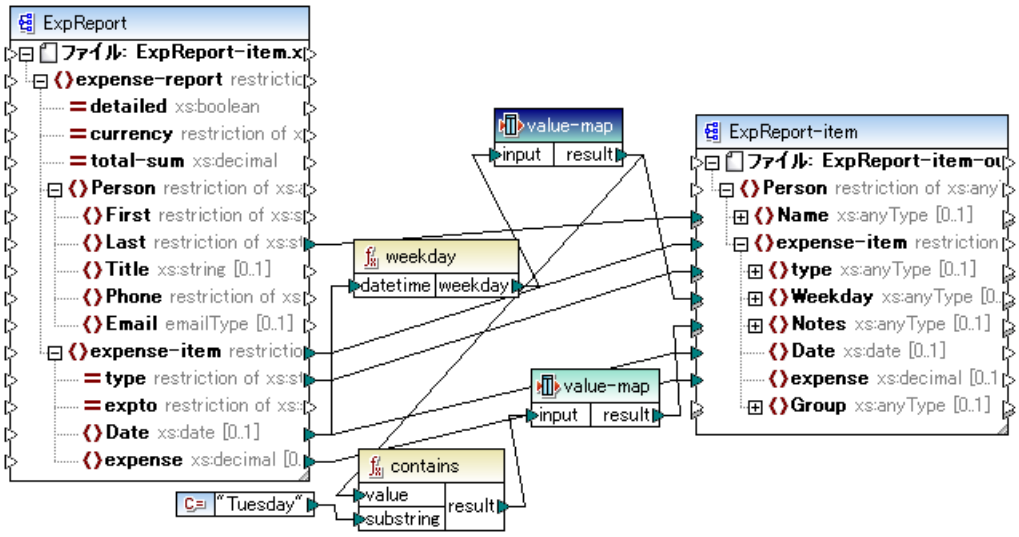
4. 変換の対象となる値を**Weekday input** カラムに入力します。
5. 変換後の値を**Day of the week** カラムに入力します。
6. 新たな値のペアを入力するには **新規エントリー)** 入力フィールドにて値を入力します。
7. カラムヘッダーの下にある**データ型**コンボボックスをクリックして、入力ならびに出力のデータ型を選択します (下の例では integer と string)。



メモ:**それ以外**のチェックボックスを選択することで、与えられた値がテーブルの値にマッチしない場合の代替出力値を定義することができます。ソースデータから与えられた値を変更すること無くパススルーさせる方法については Value-Map にて値を変更することなくパススルーさせるを参照してください。

8. ヘッダー行にある編集アイコンをクリックすることで、列名を変更することができます (列名はマッピングにも表示されます)。この操作により、マッピングにあるコンポーネントの目的を簡単に理解することができるようになります。

...\MapForceExamples\Tutorial\ フォルダーに収められている**Expense-valmap.mfd** ファイルにて、Value-Map を使用方法を示すためのサンプルマッピングを参照することができます。



このマッピングについて：
データソース内に収められているDate アイテムから、週の曜日を取得し、数値データをテキストへ変換した後、その値をターゲットコンポーネントのWeekday アイテムへ配置します。

- weekday 関数により、ExpReport ソースファイルのDate アイテムから週の曜日を取得します。この関数の戻り値は 1から 7までの数値 (nteger) となります。
- Value-Map コンポーネントにより、数値が $unday や Monday といった曜日に変換されます (上にあるスクリーンショットを参照)。
- 出力がTuesday である場合、"Prepare Financial Report" という文字列がターゲットコンポーネントの Notes アイテムへマッピングされます。
- 出力タブをクリックすることで、変換が行われたターゲットXML ファイルを確認することができます。

```
 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>
18    <expense-item>
19      <type>Lodging</type>
```

メモ：

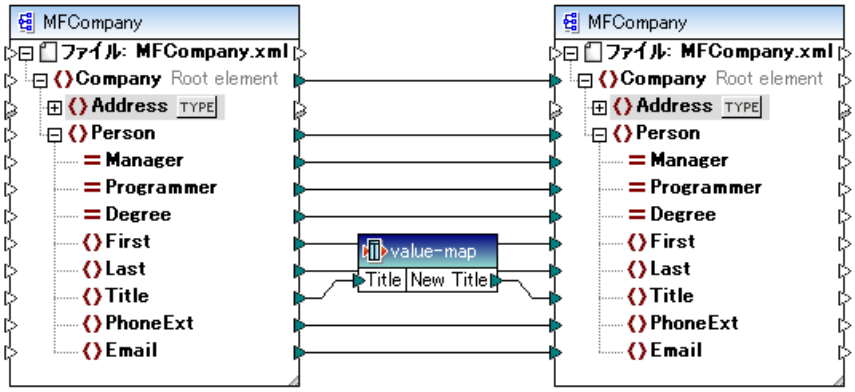
マウスカーソルをValue-Map コンポーネントの上に移動させることで、現在定義されている値を参照することができます。

論理または文字列関数から得られた出力は**"true"** または**"false"** 値となります。Value-Map テーブルの**入力**フィールドに入力する値は、例えば "true" という値になります。

5.11.1 値を変更せずに Value- Map を通過させる

このセクションでは、特定のノードデータを変換し、それ以外のノードデータをそのままのかたちでターゲットへ受け渡すといった状況について記述します。

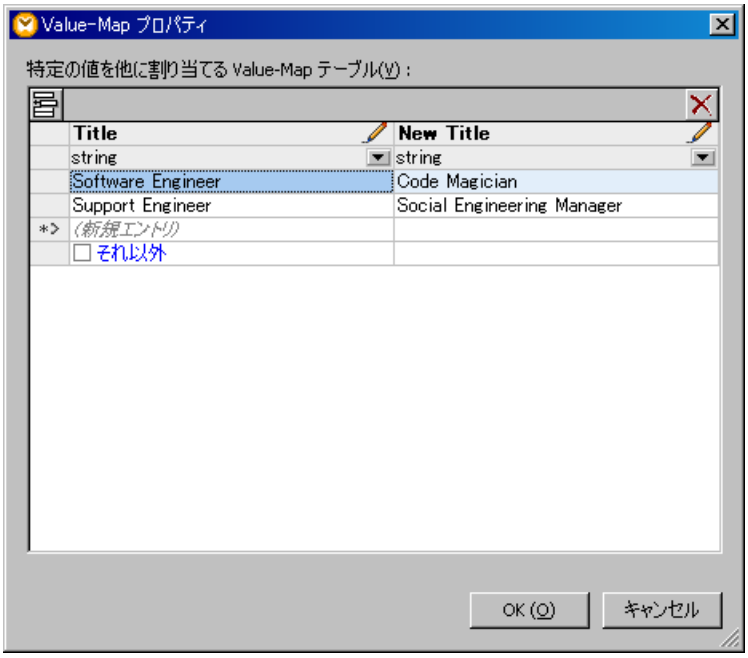
このような状況の例として、役職名を変更する会社内部の例を取り上げます。役職に関する箇所だけが変更され、それ以外はそのまま残すようにします。



マッピングは、を特定のタイトルを変換するためにvalue-map コンポーネントを使用する上のスクリーンショットのようになります。
「出力」タブをクリックすると、マッピングの結果が表示されます：

```
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>
```

Value-Map コンポーネント内にある値のどれにも当てはまらない場合、Title 要素が出力ファイルにて削除されることになります。

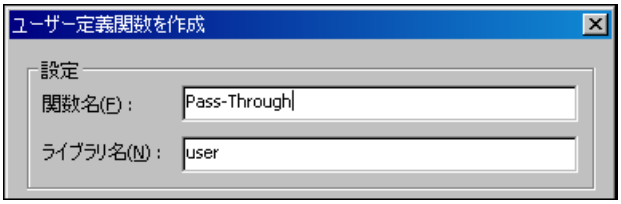


代替案：
それ以外チェックボックスをクリックして、代替となる値を入力することで、その値が出力ファイルに表示されることになります。しかしその場合、テーブルに入力されていない社内の役職全てが、**同じ値**で置き換えられてしまいます。

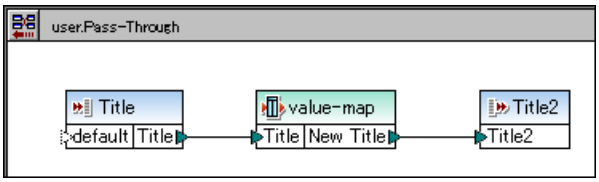
解決方法：

Value-Map コンポーネントを含むユーザー定義関数を作成し、**substitute-missing** 関数を使用することで空のノードに対してオリジナルのデータを渡します。

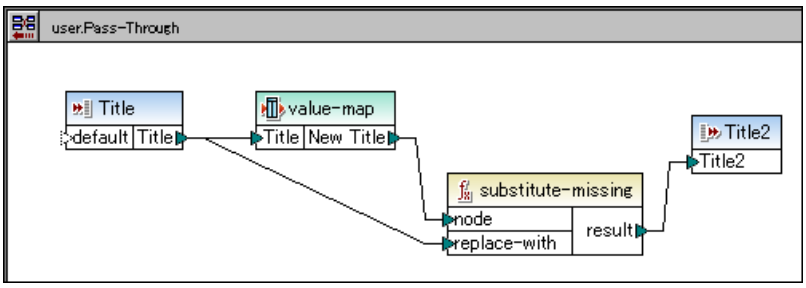
1. Value-Map コンポーネントを選択し、「関数 | 選択からユーザー定義関数を作成」を選択します。



2. 関数名にPass-Through と入力し、「**OK**」をクリックします。



3. ライブラリペインの**core | node 関数** にある**substitute-missing** 関数を挿入し、以下のスクリーンショットにあるような接続を行います。



4. 出力ボタンをクリックして結果を確認します。

マッピングの結果：

- Value-Map コンポーネントにある 2 つの役職がマッピングの結果に変換されます。
- ソースファイルにあるその他の役職が、そのままのかたちでターゲットファイルへ書きこまれます。

```
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>
```

動作の説明：
Value-Map コンポーネントは入力されたデータを評価します。

- 入力データが Value-Map の最初のカラムにある値の**どれかとマッチ**した場合、データの変換が行われ、その値が substitute-missing 関数のパラメーターへ渡され、その後 Title2 へと渡されます。

- 入力データがカラムのデータに**マッチしない**場合、Value-Map コンポーネントの出力からノードパラメーターへ値が渡されることはなく、**空のノード**が渡されます。

  空のノードが発生した場合、substitute-missing 関数によりオリジナルのノードが Title ノードから取得され、**replace-with** パラメーターへ渡された値が Title2 へと渡されます。

5.11.2 Value- Map コンポーネントプロパティ

動作：


現在アクティブな行の前に新たな行を**挿入**します。

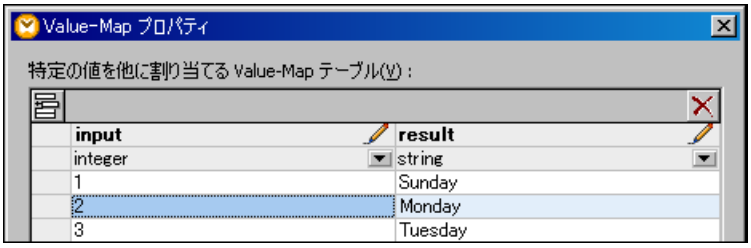

現在アクティブな行を**削除**します


カラムヘッダーの**編集**を行います。

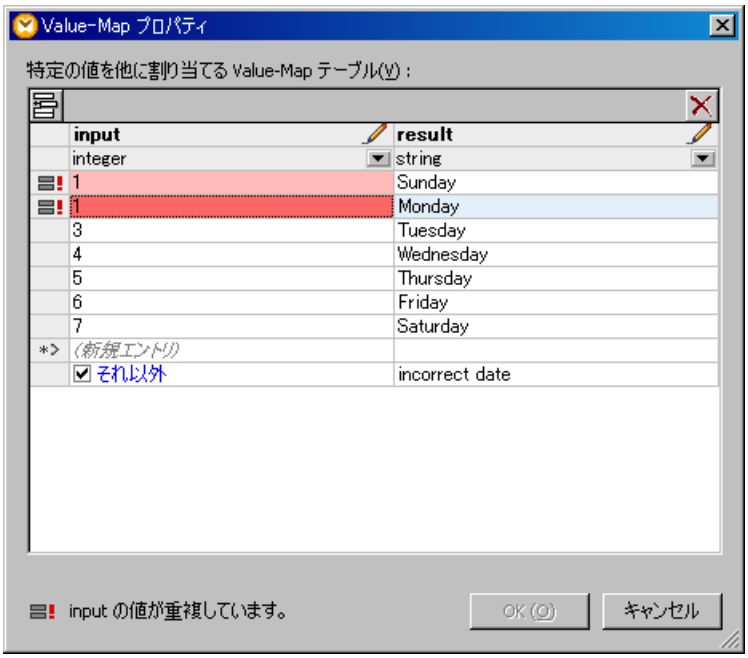
行をドラッグすることで順番を変更することもできます。

カラムのヘッダーを変更する：

カラムヘッダーをダブルクリックするか鉛筆アイコンをクリックすることで、列名を編集し、より分かりやすい名前に変更することができます。列名はマッピングにも表示されるため、この操作によりコンポーネントの目的を簡単に理解することができるようになります。



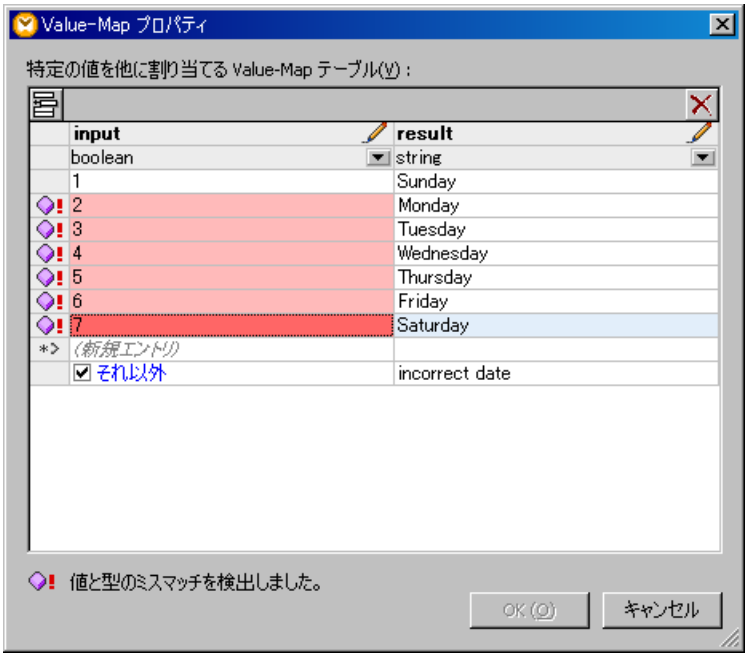
ユニークな入力値を使用する：:
入力カラムに入力される値はユニークでなければなりません。同じ値を 2 回以上入力した場合、その値がハイライトされ、修正するよう促されます。



値を修正すると、「OK」ボタンが再度選択可能になります。

入力ならびに出力データ型
コンボボックスを使ってデータ型が選択された際に、入力ならびに出力データ型が自動的にチェックされます。ミスマッチが発生した場合、該当するフィールドがハイライトされ、「OK」ボタンが選択不可能になります。値をサポートするデータ型へ変更してください。

以下のスクリーンショットでは、boolean とstring が選択されています。


Value-Map プロパティ
特定の値を他に割り当てる Value-Map テーブル(V) :
input
result
boolean
string
1
Sunday
2
Monday
3
Tuesday
4
Wednesday
5
Thursday
6
Friday
7
Saturday
*>
(新規エントリ)
それ以外
incorrect date
値と型のミスマッチを検出しました。
OK (O)
キャンセル

5.12 例外の追加

例外は、フィルターにより返された条件が発生すると、マッピングの処理を停止し、エラーを表示する特別なコンポーネントの型です。true/false 条件をチェックするフィルターをマッピングが含む場合、例外を追加することができます (次を参照 : フィルターと条件)。例えば、カスタム化されたしきい値よりマッピングアイテムの一部の値が大きい場合、例外をスローするなどが挙げられます。

MapForce を使用して、WSDL Web サービスをデザインする際にも例外を追加することができます。次を参照 : Web サービスのフォールト)。

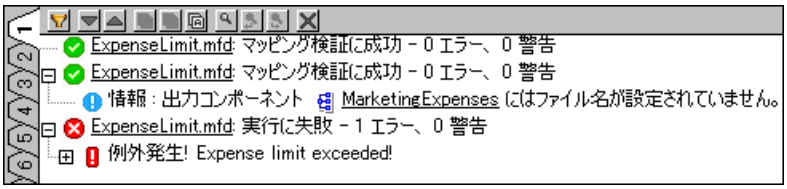
例外をマッピングに追加する :

1. **「挿入」** メニューから、 **「例外」** をクリックします。
2. **「例外の挿入」** ( ) ツールバーボタンをクリックします。
3. 例外 **throw** 入力をフィルターの **on-true** または **on-false** 出力に接続します。
4. この例外がスローされると、任意で、例外の **error-text** 入力を、エラーのテキストを与える他の (通常、定数である) コンポーネントに接続します。

メモ : フィルターの **on-true** と **on-false** 出力が双方接続されている必要があります。具体的には、これら出力の内の 1 つが、例外に (中間関数または、コンポーネントを使用することなく) 直接接続されていなければなりません。

マッピングに例外が発生すると、以下のような通知が送信されます :

- MapForce 内でメッセージウィンドウはエラーと例外テキストを表示します。 "Expense limit exceeded")。



マッピング言語が XSLT 2.0 または XQuery の場合、 「実行に失敗しました」 "エラーがメッセージウィンドウ内で表示され、対応する XSLT2 または XQuery タブが開かれます。エラーの生じたラインはメッセージウィンドウ内でハイライトされます。

- MapForce Server からマッピングを実行する場合は、MapForce 内で定義されたカスタムの例外テキストに続きエラー 『例外がスローされました』 が返されます。

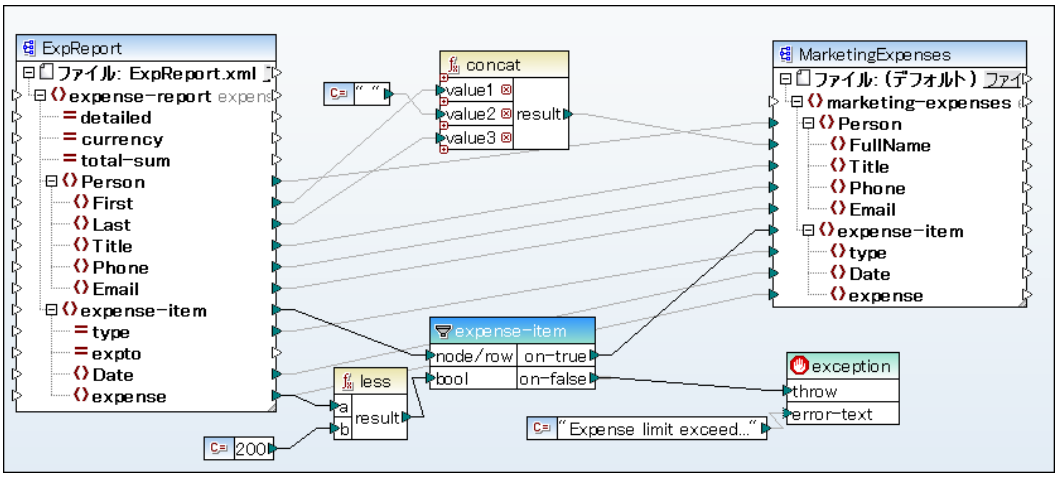
```
c:\codegen\mfx\ExpenseLimit>"C:\Program Files (x86)\Altova\MapForceServer2016\bi
n\MapForceServer.exe" run ExpenseLimit.mfx
Exception was thrown! Expense limit exceeded!
```

- 生成された C#、C++、または Java コードからマッピングを実行すると、エラー "USER EXCEPTION" が、内で定義されたカスタム例外テキストの前に返されます。

```
C:\codegen\cs\ExpenseLimit\Mapping\bin\Debug>Mapping.exe
Mapping Application
USER EXCEPTION: Expense limit exceeded!
```

5.12.1 例 :"より大きい" 条件の例外

この例では、「より大きい」条件が発生すると例外を発生するマッピングの作成方法を説明します。このマッピングのためのサンプルマッピングは以下にあります : **<Documents>\Altova\MapForce2017\MapForceExamples\ExpenseLimit.mfd**。



このマッピングは、ソースXML インスタンス内の**expense** アイテムが 200以上の値を有すると、例外をスローします。値 "200" は、定数により与えられます。**less** 関数 は、2つの値を比較するために使用されます。 **expense** の値が 200 より小さい場合、その親である**expense-item** をフィルターにパスされ、例外はスローされません。それ以外の場合、カスタムテキスト"Expense limit exceed" と共に、例外がスローされます。

上に示されるように、例外は、 アイコンにより識別され、次の2つのアイテムより構成されます :**throw** と**error-text**。 **throw** アイテムは、フィルターの**on-false** または**on-true** 出力に接続されている必要があります。**error-text** は、例外のカスタムテキストを与える定数に接続されています。

重要な点は、フィルターの両方の出力は接続されています。それ以外の場合、例外は、スローされません。この特別な例では、**on-false** 出力は、例外に接続されていますが、**on-true** 出力は、ターゲットコンポーネントに接続されています。

5.12.2 例 :ノードが存在しない場合の例外

この例では、ソースXML スキーマが存在しない場合例外をスローする方法を説明します。説明を簡素化するために、この例は、同じXML スキーマをソースとターゲットコンポーネントで使用します。

ソーススキーマをマッピングに追加する:

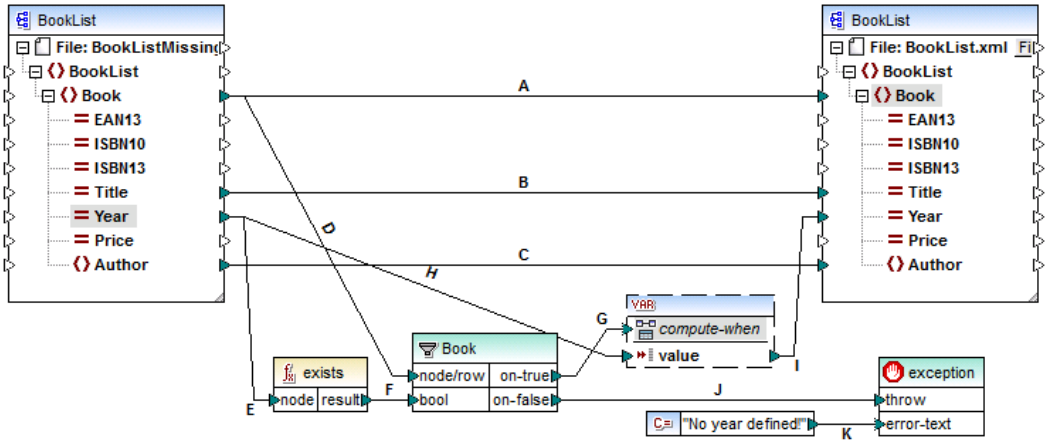
1. **「挿入」** メニューから **「XML スキーマ/ファイル」** をクリックし**<Documents>\Altova\MapForce2017\MapForceExamples\BookList.xsd** を参照します。
2. インスタンスファイルを提供するようにプロンプトされると、 **「スキップ」** をクリックします。
3. スキーマルート要素を選択するようにプロンプトされると、**BookList** をルート要素として選択します。

ターゲットスキーマを追加するには、同じステップに従います。そして、対応するコマンド (または、対応するツールバーボタン) を **「挿入」** メニューから使用し、以下を追加します :

- **Filter: Nodes/Rows** コンポーネント (以下も参照してください : フィルターと条件)
- 単純型 "文字列" の中間変数

- テキスト "No year defined!" を伴う定数
- 例外

最後に、**ライブラリ**ウィンドウから、`exists` 関数をマッピングエリアにドラッグして、下に表示されるように接続を描きます。



XML スキーマに従うと、書籍の名前以外の**Book** 要素の全ての属性は、任意です。ですから、"Year" 要素は、有効な XML インスタンス内に存在する、もしくは、しない場合があります。このマッピングのゴールは、各書籍のために "Year" 要素が存在する箇所で XML インスタンスを正確に処理することです。それ以外の場合、マッピングは、例外をスローします。

マッピングの例外の成功をテストする:

1. ソースコンポーネントのヘッダーをダブルクリックします。 **「入力 XML ファイル」**の横から、次のファイルを参照します: **<Documents>\Altova\MapForce2017\MapForceExamples\BookList.xml**.
2. **「出力」**ボタンをクリックしてマッピングを実行します。

例外をテストする:

1. 同じディレクトリ内に、**BookListInvalid.xml** と呼ばれる**BookList.xml** ファイルのコピーを作成します。
2. Book 要素から "Year" 要素を削除するように修正します。
3. ソースコンポーネントのヘッダーをダブルクリックします。 **「入力 XML ファイル」**の横から、**BookListInvalid.xml** ファイルを参照します。
4. **「出力」**ボタンをクリックしてマッピングを実行します。

マッピングの作動について説明します。前の例とは異なり、このマッピングは、中間変数を使用します。(以下も参照してください: 変数の使用)。中間変数の目的は、存在する場合、"Year" 要素の値を一時的に各書籍のコンテキストに保管し、ターゲットファイルに書き込むことです。 "Year" 値が不足している可能性があるため、この場合を確認するためにも、ソースコンポーネントの**Year** 出力は、意図的にターゲットコンポーネントの**Year** 入力に接続されていません。

接続 **A** は、ソースインスタンス内に各書籍のためにターゲットインスタンス内に書籍が作成されることを確認します。接続 **B** と**C** は、 "Title" と "Author" がソースからターゲットに、それぞれの書籍で、コピーされていることを保証します。

接続 **D** は、**Book** をフィルターに条件付きでパスします。具体的には、フィルターは、"Year" 要素を持つ書籍のみを抽出します。これを行うには、接続 **E** は、`exists` 関数 "Year" 要素 の存在をチェックするようにトリガーされます。接続 **F** は、関数の結果 (`true` または `false`) をフィルターにパスします。結果が `true` の場合、 "Year" 要素は、存在し、書籍はフィルターにパスされます。また、変数が接続 **H** を使用して 年度の値により作成されます。

> ソースコンポーネントの**Year** 出力にフィルターが直接接続されていないことに注意してください。これを行った場合、フィ

ルターは自身の存在により**Year** をフィルターするので、意味が無く、例外は決してスローされません。

接続 **G** は、フィルターの条件がtrue の場合、変数が計算されることを保証します。 (すなわち、"Year" 要素 が存在する都度、変数が計算されます)。記述の通り、ソースコンポーネントの**Year** 出力からの接続 **H** により変数の値が与えられます。接続 **I** は、値をターゲットコンポーネントにパスします。

ルールに従い、例外は、フィルターの**on-false** または**on-true** 出力に接続されていなければならないため、接続 **J** が存在します。最後に、接続 **K** は、カスタムエラーテキストを定数から例外コンポーネントにパスします。

5.13 文字列の解析とシリアル化

文字列の解析とシリアル化は、文字列からデータを解析、またはデータを文字列にシリアル化するコンポーネントを構成する高度なマッピング技術です。この技術は、ファイルからデータを読み込む(またはファイルにデータを書き込む)代替として使用することができます。文字列を解析する、またはデータを文字列にシリアル化するMapForce コンポーネントは、以下のようなシナリオにとても役にたちます。例:

- XML、テキスト、またはJSON ファイル などの構造をデータベースフィールド またはExcel スプレッドシートセルに挿入する必要がある場合。
- データベースフィールド内に保管されたXML フラグメントを スタンドアロンのXML ファイルに変換する必要がある場合。
- (例えば、FlexText を使用する)レガシーデータをテキスト (例:単一のデータベースフィールド内の固定値コンテンツ)として保存しており、このデータを並べ替え可能なフィールドベースの構造 にするために変換する場合。

文字列の解析とシリアル化 以下のMapForce コンポーネント型で使用することができます:

- テキスト (CSV、固定値フィールド テキスト、EDI、とMapForce Flexテキスト テンプレート)
- JSON スキーマファイル
- XML スキーマファイル

文字列の解析とシリアル化は、以下のMapForce ターゲット 言語でサポートされています。

言語	読み込み	書き込み
BUILT-IN (マッピング変換のプレビュー)	はい	はい
BUILT-IN (MapForce Server 実行可能ファイルを実行)	はい	はい

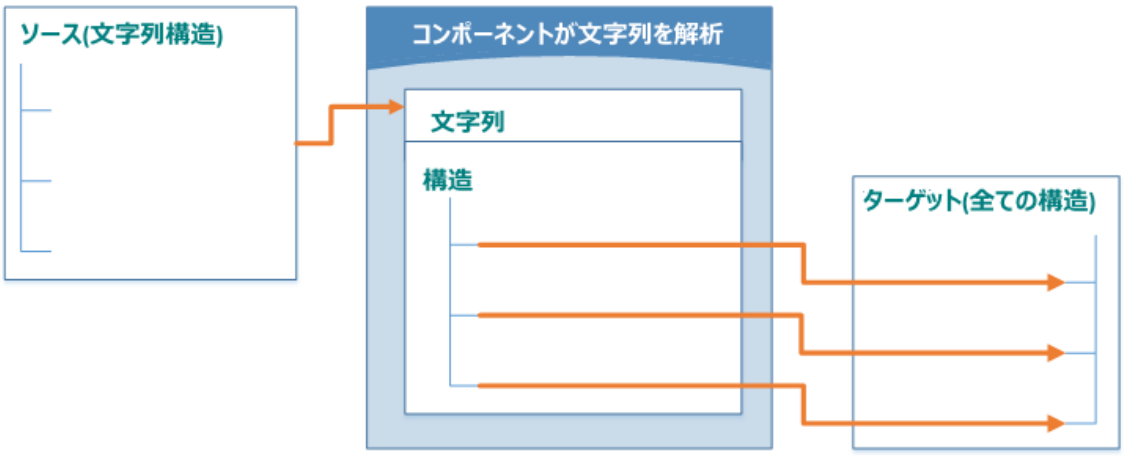
このセクションは、以下のトピックを含みます:

- コンポーネントの解析 /シリアル化について
- 例:文字列の解析 (固定長テキストからExcel へ)
- 例:文字列にシリアル化 (XML からデータベースへ)

5.13.1 コンポーネントの解析 /シリアル化について

MapForce 内のコンポーネントの解析 /シリアル化 コンポーネントは、ソースもしくはターゲットコンポーネントではない、ハイブリッドのコンポーネントです。マッピングデザイン内での役割を考えると、このようなコンポーネントはソースとターゲットコンポーネントの間に置かれることが適切です。

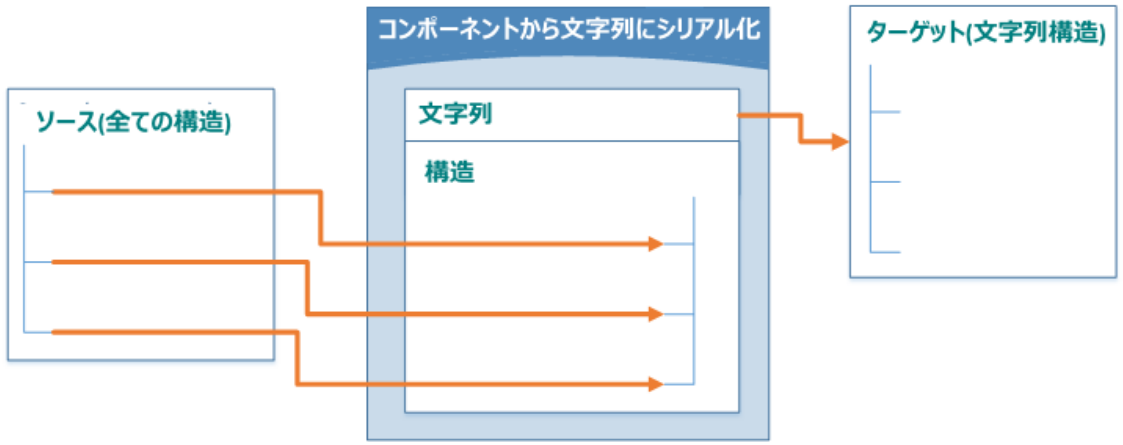
文字列の解析ためのコンポーネントの "文字列の解析 /シリアル化 "のためには、構造を持つ文字列を他のフォーマットに変換する必要があります (例:データベース内に文字列として保管されているXML)。ソース文字列からデータを解析するには、コンポーネントの "解析 /シリアル化 "は、ソース文字列がMapForce 構造に変換されることを意味します。ですから、文字列として保管されたソースXML の要素または属性へのアクセスを取得することができます。



ジェネリックな "文字列の解析 "コンポーネント

上の図は、MapForce コンポーネントが文字列を解析する全般的な構造を示しています。コンポーネントの "文字列の解析 シリアル化 "は、マッピングのソースとターゲットの間に配置されています。このコンポーネントは、文字列の構造の一部をトップ**文字列**ノードに接続されている単一のMapForce コネクタを入力として受け入れません。出力構造は、MapForce によりサポートされるデータターゲットとなることができます。例えば、 例 :文字列の解析 固定長テキストを Excel へ)を参照。

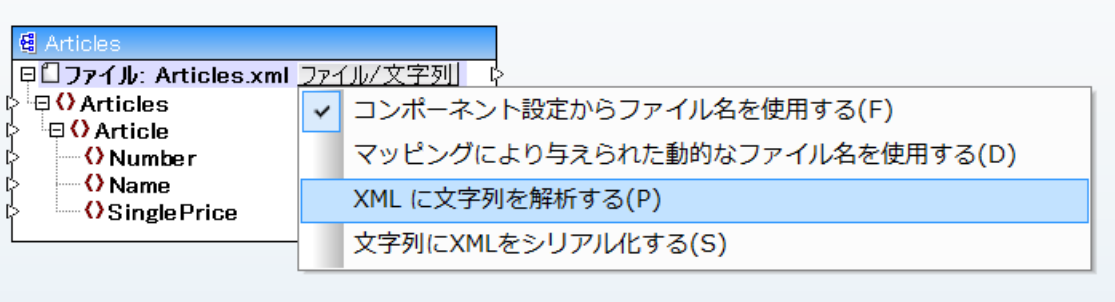
コンポーネントから文字列にデータをシリアル化すると、逆の効果が発生します。特に、MapForce コンポーネントの全体の構造が、必要であれば更に操作することが可能な文字列 構造になります。 例 :これによりXML ファイル (またはXML フラグメント)から、データベースフィールドまたはExcel スプレッドシートの単一のセルに書き込むことが可能になります。



ジェネリックな "文字列にシリアル化 "コンポーネント

上の図は、MapForce でのジェネリックなコンポーネントの"文字列にシリアル化 "を示しています。このコンポーネントは (標準 MapForce コネクタを使用して)MapForce によりサポートされるデータソースを入力として受け入れます。出力構造は、コンポーネントのトップ**文字列**からターゲットコンポーネントアイテムに描かれた単一のMapForce コネクタを使用して、更にパスすることのできる文字列 (例 :スプレッドシートセル)です。 例 :文字列にシリアル化 (XML からデータベース へ)を参照してください。

文字列の解析またはシリアル化のために、コンポーネントをマッピングウィンドウからいつでも指定することができます。これを行うには、ルートノード横の **「ファイル/文字列」** (File/String) ボタンをクリックして、希望するオプションを選択します。



コンポーネントモードの変更

メモ: "文字列の解析 /シリアル化 "コンポーネントは、文字列からデータを読み込み、同時に文字列に書き込むことはできません。ですから、ルートノードは入力コネクタ、または出力コネクタの (両方ではなく)どちらかを持つことができます。両方のオペレーションに同じコンポーネントを使用しようとすると、エラーが生成されます。

コンポーネントを文字列の解析またはシリアル化に指定する場合、コンポーネントの外観の変更は以下の通りです :

- コンポーネントはタイトル内に**解析**または**シリアル化**のプレフィックスを与えられます。
- タイトルバーは関数コンポーネントに類似した黄色の背景色が与えられます。
- トップのノードは **文字列:** プレフィックスから始まり、 アイコンで識別されます。
- コンポーネントが文字列を解析する場合、ルートノードからの出力コネクタは有効ではなく 使用することはできません。
- コンポーネントが文字列にシリアル化される場合、ルートノードへの入力コネクタは、有効ではなく使用することはできません。

コンポーネントが "文字列の解析 /シリアル化 "モード内にある場合、ファイルベースのモードと同様の法要で設定を変更することができます。 (コンポーネント設定を変更するを参照)。 コンポーネントが "解析 "または "シリアル化 "モードの場合全てのコンポーネント設定を使用できない場合があります。

5.13.2 サンプル :文字列の解析　固定長テキストから Excel へ)

このサンプルは、文字列データを解析するマッピングデザインの方法について説明します。このサンプルにはサンプルファイルが存在します。このサンプルを開始する前に、サンプルファイルを確認するには、ファイルを以下のパスで検索することができます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\ParseString.mfd**。

単一のデータベースフィールドに保管されているレガシーテキストデータがあると仮定します。テキストデータは従業員のリストであり、 RESOURCE 列内に保管されています。固定値としてフォーマットされたフィールドは、以下のとおりです :

```
|----|-----------------------------------------------------------------------------------------|------------|
| ID | RESOURCE                                                                                | TYPE       |
|----|-----------------------------------------------------------------------------------------|------------|
| 1  | P00001Callaby        Vernon        582Office Manager          v.callaby@nanonull.com    | Employees  |
|    | P00002Further        Frank         471Accounts Receivable     f.further@nanonull.com    |            |
|    | P00003Matise         Loby          963Accounting Manager      l.matise@nanonull.com     |            |
|    | P00004Firstbread     Joe           621Marketing Manager Europe j.firstbread@nanonull.com |            |
|    | P00005Sanna          Susi          753Art Director            s.sanna@nanonull.com       |            |
|    | P00006Landis         Fred          951Program Manager         f.landis@nanonull.com      |            |
|    | P00007Butler         Michelle      654Software Engineer       m.landis@nanonull.com      |            |
|    | P00008Little         Ted           852Software Engineer       t.little@nanonull.com      |            |
|    | P00009Way            Ann           951Technical Writer        a.way@nanonull.com         |            |
|    | P00010Gardner        Liz           753Software Engineer       l.gardner@nanonull.com     |            |
|    | P00011Smith          Paul          334Software Engineer       p.smith@nanonull.com       |            |
|----|-----------------------------------------------------------------------------------------|------------|
```

単一のデータベースフィールドにテキストデータが保管されているため、個々の従業員の記録には簡単にアクセスすることがで

きません。これにより、従業員を追加、または、削除、および並べ替えることが難しくなります。このサンプルの目的は、RESOURCE データベースフィールドからテキストデータを抽出し、記録を簡単に処理できるように構造に分割することです。

この目的は、コンポーネントの "文字列の解析 / シリアル化 "を使用することにより達成することができます。最初に、コンポーネントの "文字列の解析 / シリアル化 "はテキストデータを入力として受け入れます。そして、データを解析し、構造に変換します。最後に、ターゲットフォーマットに構造が書き込まれます。このサンプルでは、ターゲット フォーマットは Excel スプレッドシートです。かしながら、全般的に、MapForce によりサポートされる他の出力フォーマットに変換することができます。

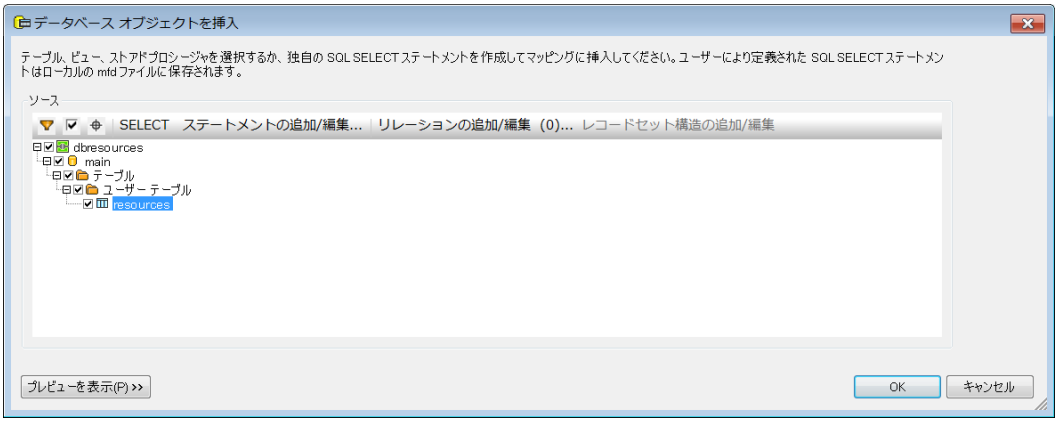
このサンプルのマッピングを要約すると、RESOURCE データベースフィールドのコンテンツをテーブルに変換します。マッピングが実行された後、各テーブル行は従業員に対応し、各列は固定長フィールドの1つに以下の順序で対応します :ID、Last Name、First Name, Extension、Job Title、Email。

	A	B	C	D	E	F	G
1	P00001	Callaby	Vernon	582	Office Manager	v.callaby@nanonull.com	
2	P00002	Further	Frank	471	Accounts Receivable	f.further@nanonull.com	
3	P00003	Matise	Loby	963	Accounting Manager	l.matise@nanonull.com	
4	P00004	Firstbread	Joe	621	Marketing Manager Europe	j.firstbread@nanonull.com	
5	P00005	Sanna	Susi	753	Art Director	s.sanna@nanonull.com	
6	P00006	Landis	Fred	951	Program Manager	f.landis@nanonull.com	
7	P00007	Butler	Michelle	654	Software Engineer	m.landis@nanonull.com	
8	P00008	Little	Ted	852	Software Engineer	t.little@nanonull.com	
9	P00009	Way	Ann	951	Technical Writer	a.way@nanonull.com	
10	P00010	Gardner	Liz	753	Software Engineer	l.gardner@nanonull.com	
11	P00011	Smith	Paul	334	Software Engineer	p.smith@nanonull.com	
12							
13							

文字列の解析後の期待される出力

目的を達成するには、以下のステップを行ってください :

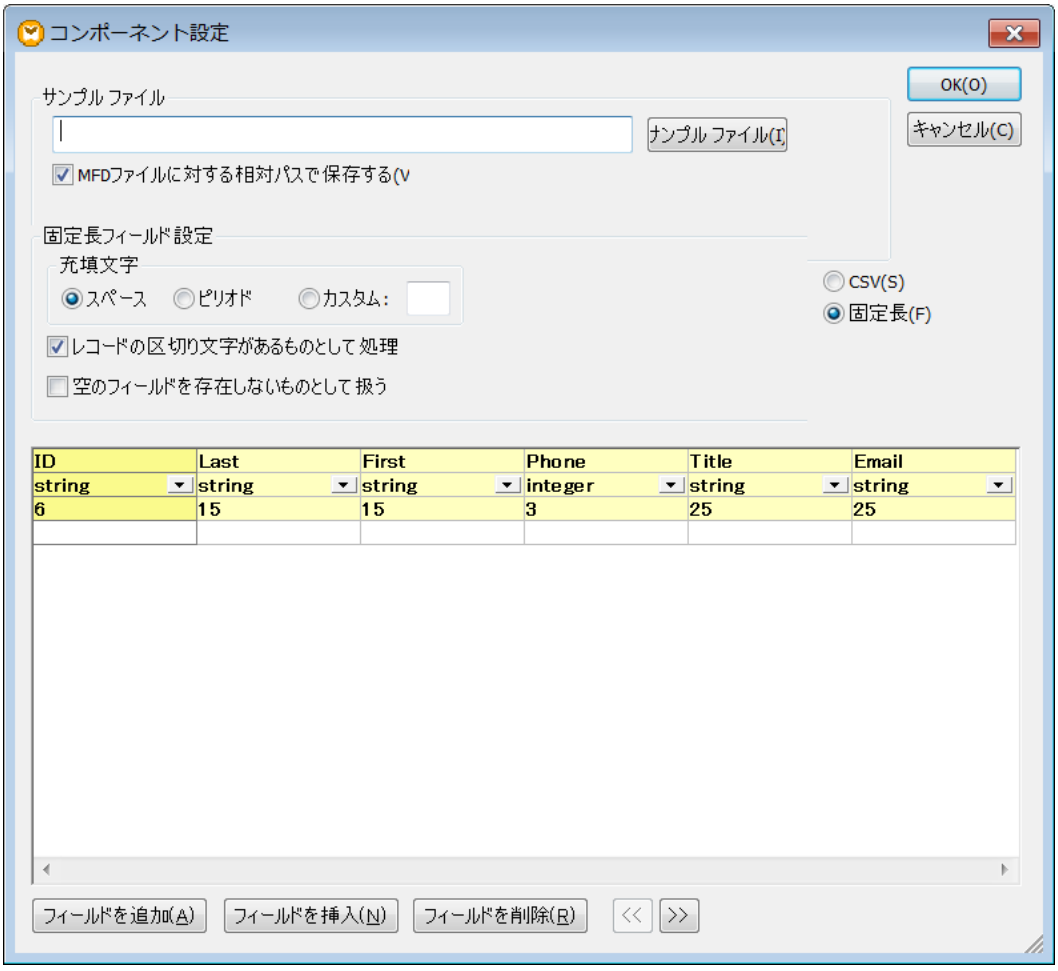
1. マッピングエリアにソースデータベースを追加します。データベースは、スタンドアロン SQLite データベースファイルとして、以下のパスから参照することができます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\\dbresources.db**。(データベースコンポーネントを追加するには、**挿入 | データベース**」メニューコマンドを使用します。 データベースへ接続するも参照)。
2. データベースオブジェクトの挿入を問われると、**resources** テーブルを選択します。



3. (**挿入 | テキストファイル**」メニューコマンドを使用して)マッピングエリアにテキストコンポーネントを追加します。ソースデータはフィールドカンマ区切りテキストであるため、MapForce によりプロンプトされると、**簡単な処理**を選択します。
4. データベース内に保管されたソーステキストデータの構造をマップするために、テキストコンポーネントの構造を構成します。既に気が付いているかもしれませんが、このサンプルでは、ソーステキストは6つの固定長フィールドから以下のように構成されています :

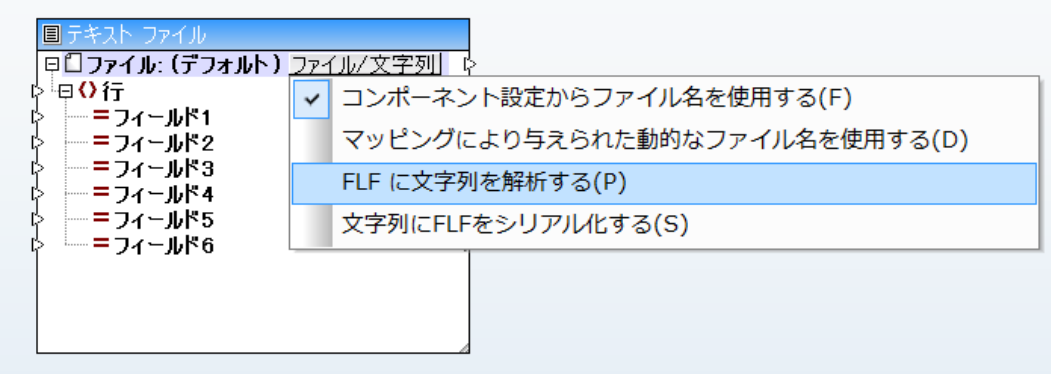
文字サイズ	フィールド
6	ID
15	Surname
15	First Name
3	Extension
25	Job Title
25	Email

これを達成するには、テキストコンポーネントを**固定長**として宣言し、上記のポジションに適用することのできる6つのフィールドを追加します。(コンポーネント設定ダイアログボックスから開くには、コンポーネントを右クリックして、コンテキストメニューから、**「プロパティ」**を選択します)。

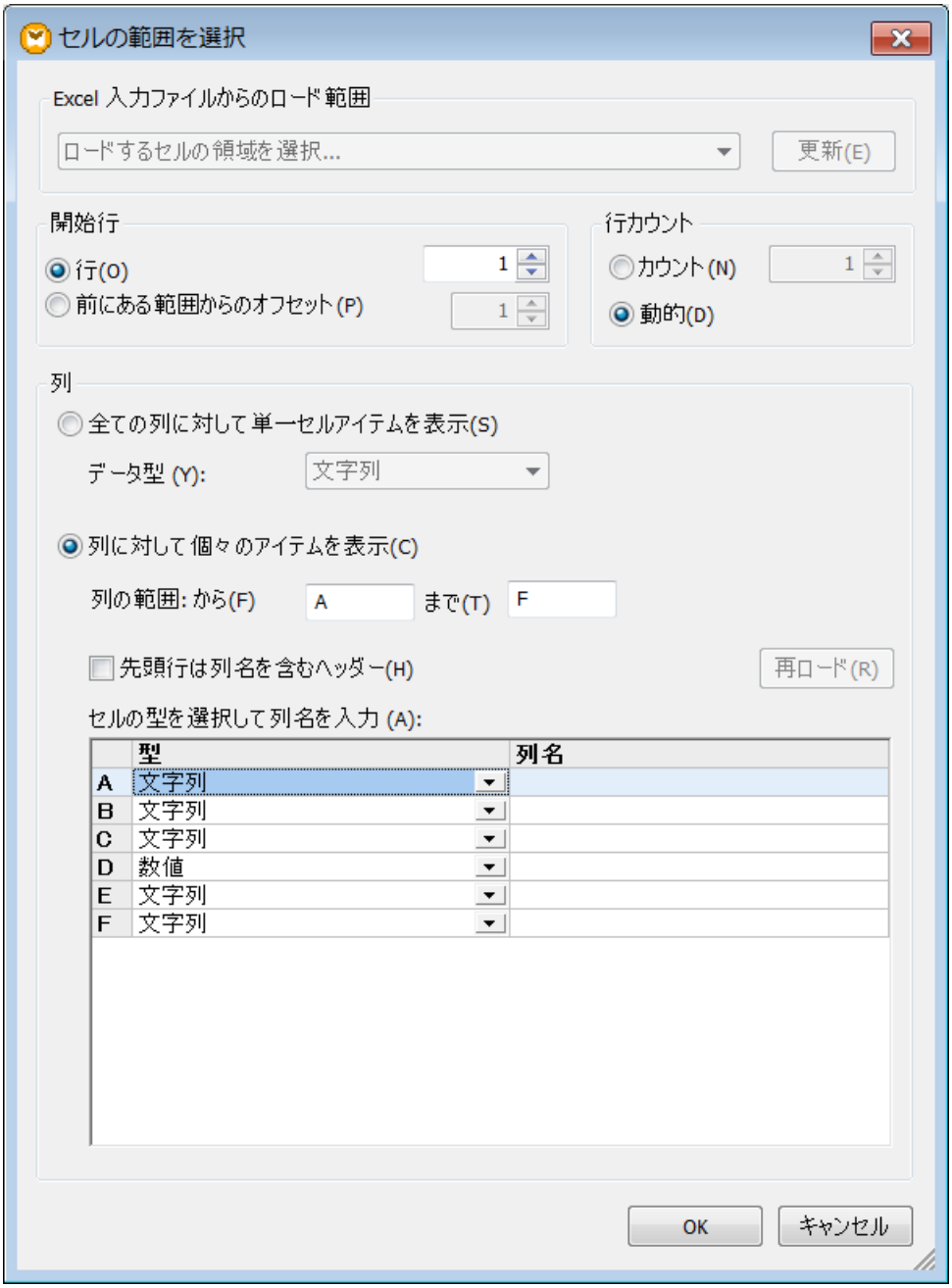


整数をデータ型として使用する**Field4** フィールドに注意してください。このサンプルのスコープのためには、**Field4** を数値型として宣言することは任意ですが、内線番号 (**Field4**) が抽出するソーステキストが数値として検証されることを確認することができます。

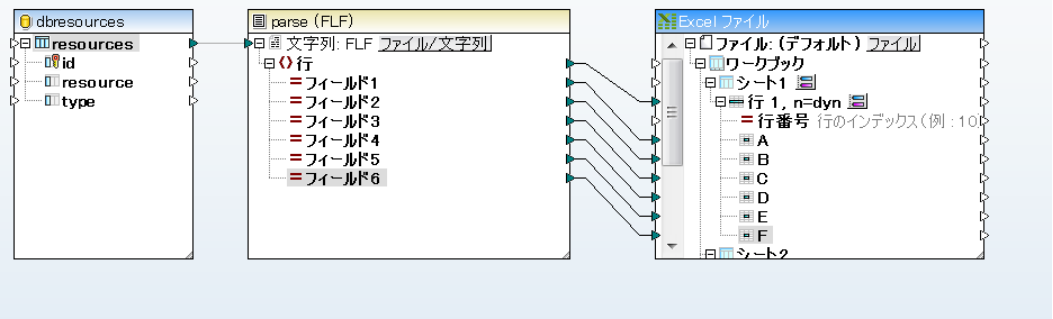
5. File/String をクリックし、**「FLF に文字列を解析する」**を「コンテキスト」メニューから選択します。これは、このコンポーネントに文字列を固定値フィールド (FLF) フォーマットに解析するように、MapForce に命令します。



6. マッピングエリアにターゲット Excel 2007+ コンポーネントを挿入します。サンプルファイルを選択するようにプロンプトされると、**【スキップ】**をクリックします。(**挿入 | Excel 2007+】**メニューコマンドを使用して Excel コンポーネントを追加することができます。Excel 2007+ ファイルをマッピングコンポーネントとして追加するも参照)。
7. **Row 1, n=dyn** ノードの横の ボタンをクリックし、Excel コンポーネントが、最初の列から各テキストフィールドのために行を書き込むことを構成します。(Excel 2007+ コンポーネント型とその構成 に関する詳細は、Excel 2007+ コンポーネントについて を参照してください)。



8. コンポーネントアイテム間の接続の描き方は以下の通りです。



マッピングの左側では、リソースデータベース列のコンテンツが文字列の値からMapForce 構造に変換 (解析) されます。マッピングの右側では、解析(*テキストファイル*)コンポーネントのアイテムが個別のExcel 列に接続され、この結果、ソース文字列を個別の並べ替えることのできるセルに分割します。

文字列データを解析し、構造を作成するMapForce デザインファイルが作成されました。 **出力** タブをクリックすると、レガシーテキストデータがExcel スプレッドシートの個別の行と列に変換されます。

5.13.3 サンプル :文字列にシリアル化 (XML からデータベースへ)

このサンプルは、データを文字列にシリアル化するマッピングデザインの作成に必要なステップを説明します。このサンプルはにはサンプルファイルが存在します。このサンプルを開始する前に、サンプルファイルを確認するには、ファイルを以下のパスで検索することができます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\SerializeToString.mfd**。

複数の<Person> 要素により構成されるXML ファイル (関連したスキーマ)があると仮定します。各 <Person> 要素は、それぞれの姓、名、役職、内線番号、電子メールアドレスを表します :

```
<Person>
      <First>Joe</First>
      <Last>Firstbread</Last>
      <Title>Marketing Manager Europe</Title>
      <PhoneExt>621</PhoneExt>
      <Email>j.firstbread@nanonull.com</Email>
</Person>
```

目的は、XML ファイルから各 <Person> 要素 を抽出し、リテラルに (XML タグも含め) SQLite データベースの PEOPLE 新規データベースの記録として挿入することです。PEOPLE テーブルは、 2つの列のみを含んでいます :ID と PERSON。定義は、以下の通りです :

```
CREATE TABLE PEOPLE (ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, PERSON
TEXT);
```

マッピングが実行された後、PEOPLE テーブルは、XML ファイル内の<Person> 要素の行数と同じ数の行を持つようになります。

このゴールを達成するには、以下を行います :

1. マッピングエリアソースXML コンポーネントに (**挿入 | XML スキーマ/ファイル** メニューコマンドを使用して)追加します。サンプルファイルは以下で検索することが出来ます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\MFCompany.xml**。
2. (コピー貼り付けを使用して)XML コンポーネントを複製します。
3. 複製されたXML コンポーネントで、 File/String をクリックして、 **XML を文字列にシリアル化する** を選択しま

す。

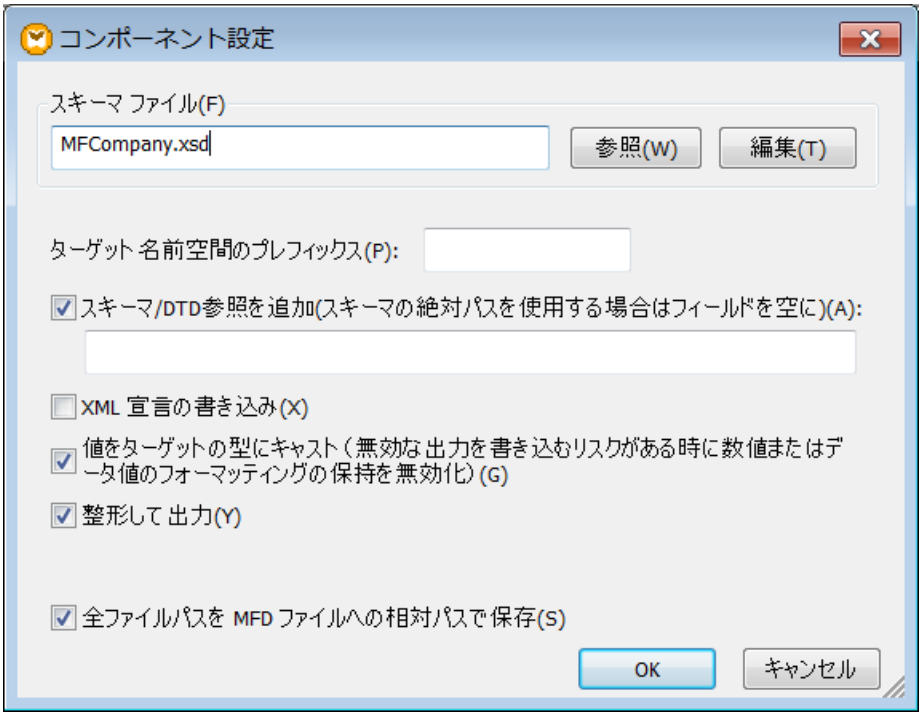


4. コンテキストメニューから、複製されたコンポーネントを右クリックして、**「ルート要素の変更」**を選択します。ルート要素を<Person> に変更します。

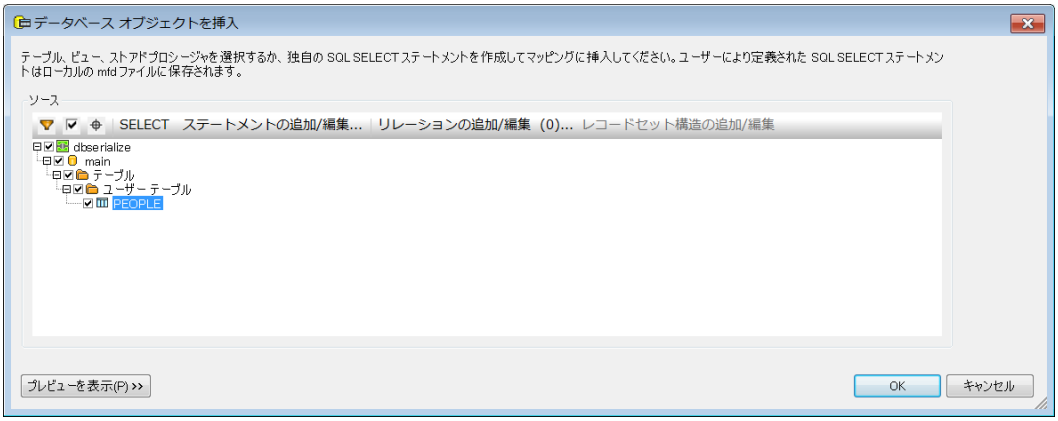


全般的に、ルート要素をXML スキーマ内で（ローカルではない）グローバル宣言を持つ要素に変更することができます。スキーマ内でグローバルに定義されていない要素は "ルート要素の選択 "ダイアログボックスにリストされません。

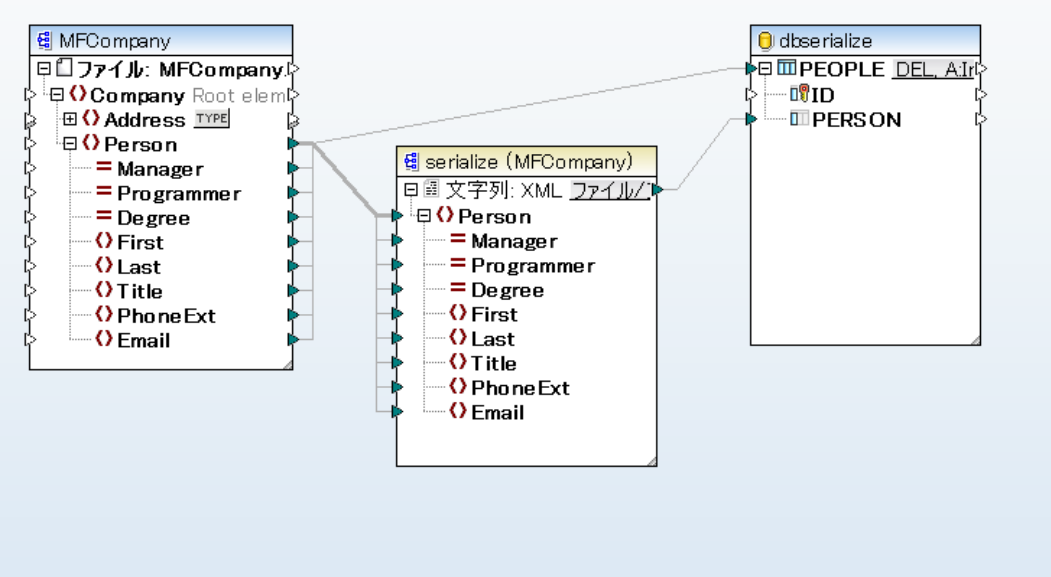
5. コンポーネントをダブルクリックし、**「XML 宣言の書き込み」**オプションを無効化します。これにより、XML 宣言が各 <Person> 要素のために書き込まれることを回避します。



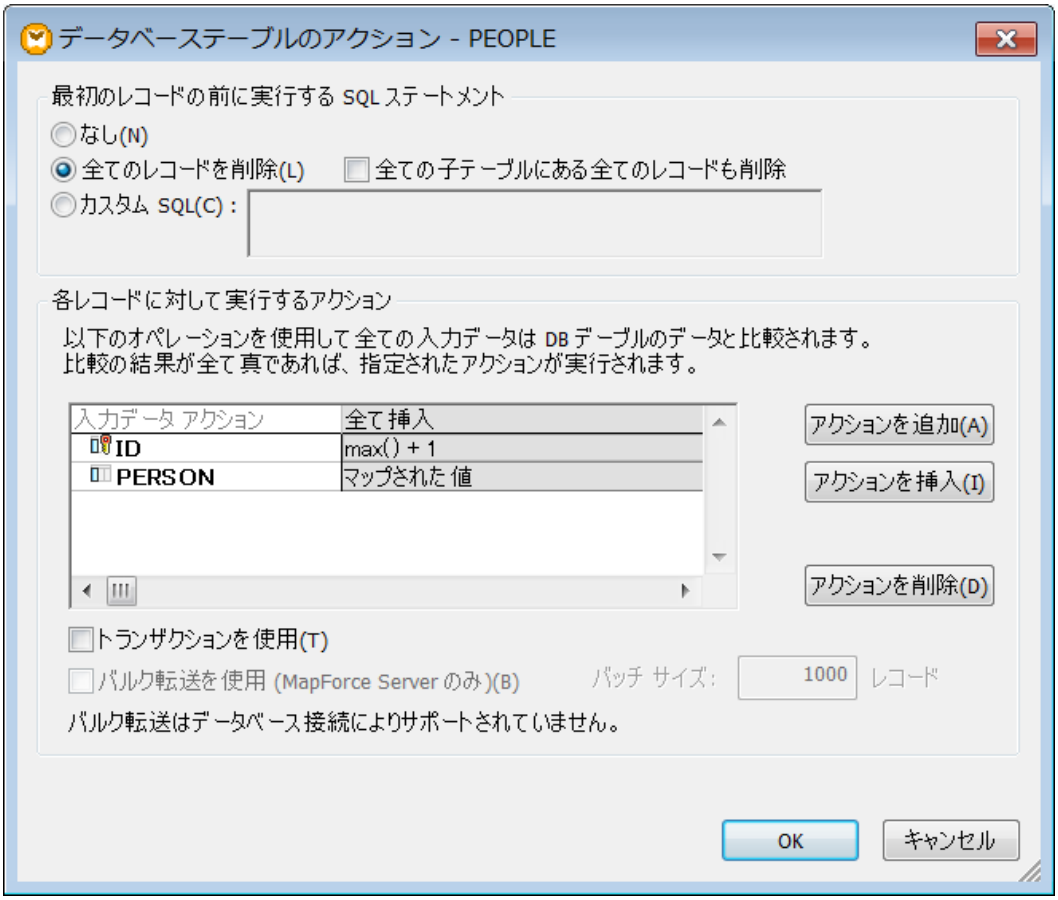
6. マッピングエリアにターゲットSQLite データベースコンポーネントを以下のパスから追加します :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\\dbserialize.db**。(**挿入 | データベース**」メニューコマンドを使用してデータベースコンポーネントを挿入します。データベースへ接続するも参照)。データベースオブジェクトを挿入するようにプロンプトされると、PEOPLE テーブルを選択します。



7. コンポーネントを下で表示されるように、リンクします。マッピングの左側では、<Person> 要素は、コンポーネントはシリアル化コンポーネントにマップされます。マッピングの右側では、PEOPLE データベーステーブルのシリアル化された文字列の値がPERSON 列に挿入されます。最後に、<Person> からPEOPLE テーブルへ描かれたコネクタは、MapForce に当てはまる各 <Person> 要素のために新しい記録を作成するように命令します。



8. データベースコンポーネントの **A:In** ボタンをクリックし、MapForce にマッピング変換が実行される都度、以下のアクションを行うように命令します：
   a. テーブルから全ての記録を削除する
   b. ID の値を 1 ずつ増分する



ID 列のために選択されていた**max()+1** アクションに注意してください。これによりデータベース内の既存のID 値の最大値をMapForce に分析するように命令します。

データを文字列にシリアル化するマッピングデザインが作成されました。 **出力」** タブをクリックすると、プレビューSQL クエリは、XML ファイル内の各 `<Person>` 要素 ために個別の記録がデータベースに挿入されることを示します。

5.14 ノード名のマッピング

MapForce を使用して、マッピングを作成すると、多くの目的はソースから値を読み取り、ターゲットに値を書き込むことです。しかしながら、ソースからだけではなく、ノード名から値を取得することを希望する場合があります。例えば、ソースXMLから(値ではなく)要素または属性の名前を読み取り、ターゲットXML内で 名前では無く要素または属性の値 に変換することを希望する場合があります。

以下の例を考慮してください：XML ファイルが製品のリストを含む場合、各製品は次のフォーマットを有します：

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

目的は、各製品の情報を名前と値のペアに変換することです。例：

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

このようなシナリオには、マッピングからのノード名へのアクセスが必要になります。このトピックの主題であるノード名への動的なアクセスにより、このようなデータ変換を行うことができます。

メモ： node-name と static-node-name コアライブラリ関数を使用して、上記の変換を行うことができます。しかしながら、この場合は、ソースから期待する正確な要素名が必要であり、ターゲットに手動で要素を接続する必要があります。また、これらの関数は、十分ではない場合があります。例えば、名前別にフィルター、または、グループ分けを行う場合、また、ノードマッピングからデータ型を操作する場合などが挙げられます。

ノード名への動的なアクセスは、ノード名を読み取る場合だけではなく、書き込む必要がある場合にも使用することができます。標準マッピングでは、基となるコンポーネントのスキーマからくる、ターゲット内の属性または要素の名前は、マッピングの実行前に既知です。動的なノード名を使用して、マッピングの実行前に既知ではない、新規の属性または要素を作成することができます。具体的には、属性または要素の名前が、MapForce によりサポートされるソースからマッピング自身により与えられます。

ノードの子要素、または、属性への動的なアクセスを可能にするには、ノードが実際に子要素または属性を持つ必要があり、XML ルートノードであってはなりません。

動的なノード名は、次のコンポーネント型に、または、型からマッピングをする際にサポートされています：

- XML
- CSV/FLF*

* MapForce Professional または Enterprise Edition が必要です。

メモ： CSV/ FLF の場合、CSV/FLF 構造には、ノードが存在しないため、動的なアクセスは、ノードの代わりにフィールドへのアクセスを意味します。

マッピングのターゲットがCSV またはFLF (固定長フィールド) ファイルである場合、フィールドは、コンポーネント設定内で定義されている必要があります (名前、順序、またはターゲットフィールドの数量を変更することはできません)。XML とは異なりテキストファイルのフォーマットは固定されており、実際のフィールドの値のみを操作することができますが、フィールド名、数量、順序を操作することはできません。

ノード名への動的なアクセスは次のマッピング言語によりサポートされています :BUILT-IN、XSLT2、XQuery、C#、C++、Java。*

* MapForce Professional またはEnterprise Edition が必要です。

ノードへの動的なアクセスに関する情報は、以下を参照してください : ノード名へのアクセスを取得する。順序を追ったマッピングの例に関しては、以下を参照してください :例 :要素名を属性値にマップする。

5.14.1 ノード名へのアクセスを取得する

XML コンポーネント (または、CSV/FLF コンポーネント内のフィールド) 内のノードに子ノードが存在する場合、マッピングの名前と各子ノードの値をマッピングで直接取得することができます。このテクニックは、「動的なノード名」へのアクセスと呼ばれます。「動的」とは、処理がランタイム中に素早く行われることを意味し、マッピングが実行される前に既知の静的なスキーマ情報をベースに行われます。このトピックは、ノード名への動的なアクセスの有効化の方法と、その使用方法について説明します。

ソースからデータを読み取る場合、「動的なノード名」とは以下を行うことができることを意味します :

- ノードの全ての子ノード (または属性) のリストをシーケンスとして取得します。MapForce では、"シーケンス" は、ゼロのリスト、または、ターゲットに接続することのできるアイテムであり、ソース内にアイテムが存在すると、ターゲットに同じ数のアイテムを作成します。ですから、例えば、ノードがソース内に5つの属性 を持つ場合、ターゲット内に属性に対応する5つの新規の要素を作成します。
- (標準マッピングが行う) 子ノードの値を読み取るだけではなく、名前も読み込みます。

ターゲットにデータを書き込む場合、「動的なノード名」は、以下を行うことができることを意味します :

- コンポーネント設定 (いわゆる「静的」名前) により与えられた名前とは対照的な、マッピング (いわゆる「動的」名前) により与えられる名前を使用して、新しいノードを作成します。

動的なノード名を説明するために、このトピックは次のXML スキーマを使用しています :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Products.xsd**。このスキーマは、サンプルインスタンスドキュメント**Products.xml** を伴っています。スキーマとインスタンスファイルをマッピングエリアに追加するには、**「挿入」| XML スキーマ/ファイル...** メニューコマンドを選択して、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Products.xml** を参照してください。ルート要素を選択するように問われると、`products` を選択します。

`product` ノードのための動的なノード名を有効化するには、右クリックし、次のコンテキストメニューコマンドの1つを選択します :

- ノードの属性にアクセスする場合、**「動的な名前を持つ属性を表示」**
- ノードの子要素にアクセスする場合、**「動的な名前を持つ子要素を表示」**

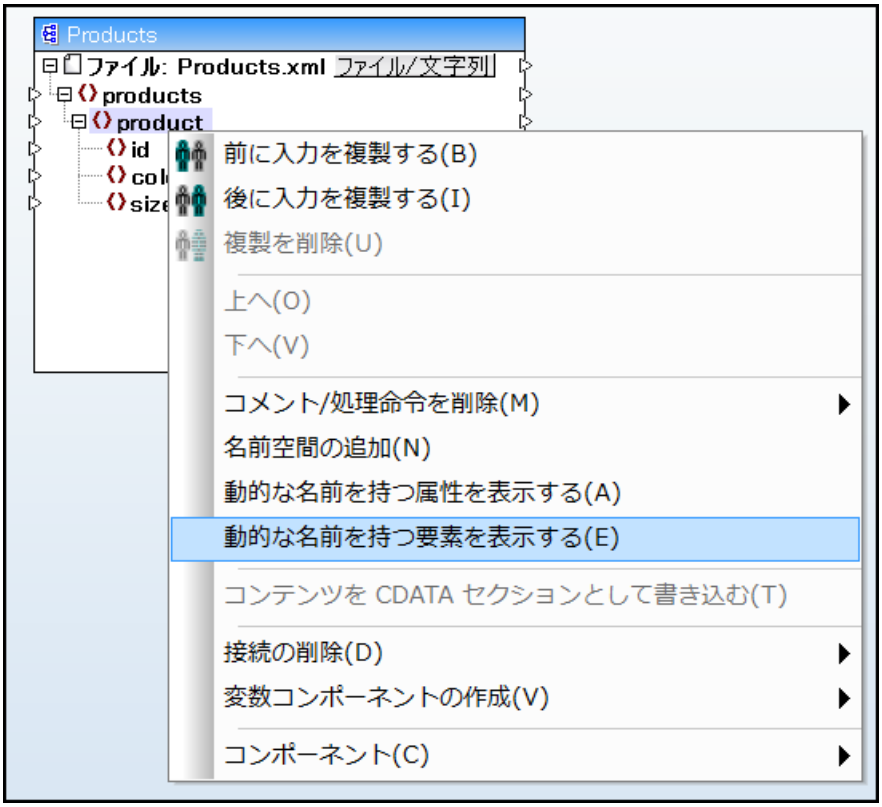


図 .1　(子要素のための)動的なノード名の有効化

メモ: 上のコマンドは、子ノードを持つノードのみに対して使用することができます。また、コマンドはフットノードには使用することができません。

ノードを動的なモードに切り替えると、下に表示されるダイアログボックスに類似したダイアログボックスが表示されます。このトピックの目的のために、オプションを下に表示されるように設定します。これらのオプションは特定の型のノードにアクセスする内で説明されています。

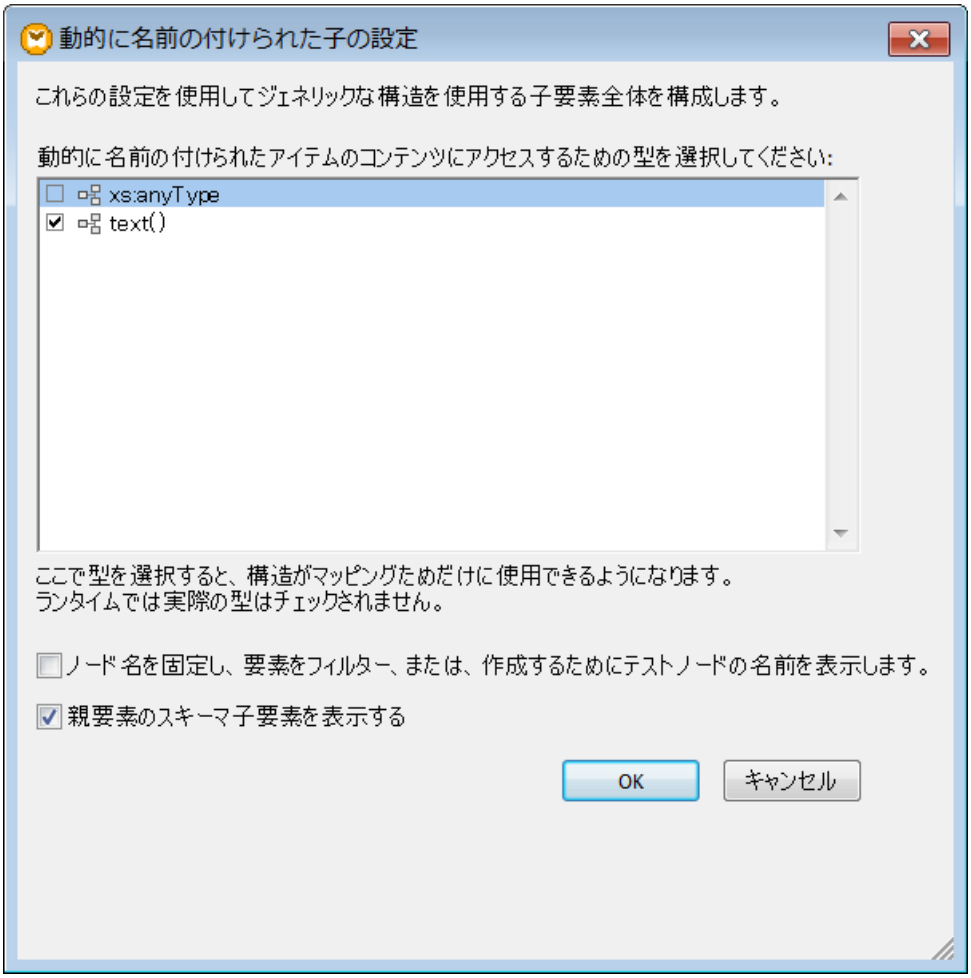


図 2 「動的に名前が付けられた子の設定」ダイアログボックス

図 3は、product ノードのために動的なノード名が有効化されている場合、どのようにコンポーネントが表示されるかを示しています。コンポーネントの外観が大幅に変更されていることにご注意してください。

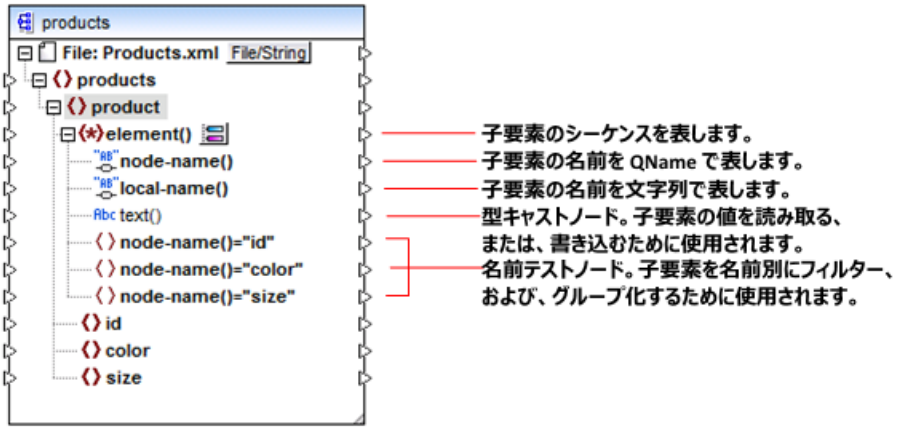


図 3 (要素のために)動的なノード名が有効化されている場合

コンポーネントを標準モードに切り替えるには、product ノードを右クリックして、コンテキストメニューからオプション **「動的な名前を持つ子要素を表示する」** を無効化します。

下のイメージは、ノードの属性への動的なアクセスが有効化された時、同じコンポーネントがどのように表示されるかを示しています。product 要素を右クリックして、コンポーネントが取得され、コンテキストメニューから **「動的な名前を持つ属性を表示する」** を選択します。

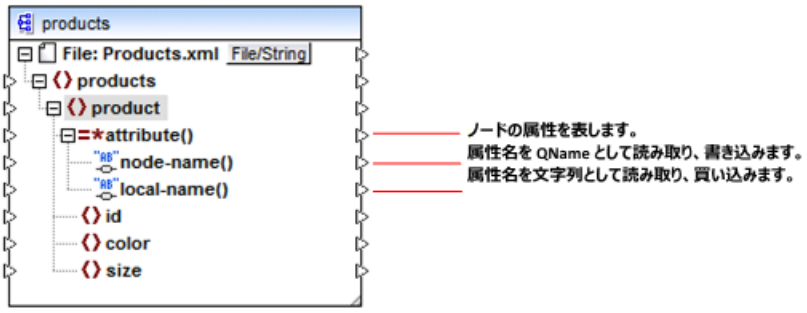


図 4　(属性のために)動的なノード名が有効化されている場合

コンポーネントを標準モードに切り替えるには、product ノードを右クリックして、コンテキストメニューから、 **「動的な名前を持つ属性を表示する」** オプションを無効化します。

図 3と図 4に示されているように、コンポーネントの外観はノード (この場合は、product) が「動的なノード名」なモードに切り替えられると変更されます。新しい外観は、次のアクションを取る可能性を広げます：

- すべてのノードの子要素または属性のリストを読み取り、または書き込みます。これらは、element() または attribute() アイテムによりそれぞれ提供されます。
- 各子要素または属性の名前を読み取り、または書き込むます。名前は、node-name() とlocal-name() アイテムにより提供されます。
- 要素の場合、各子要素の値を特有のデータ型として読み取る、または書き込みます。この値は、型キャストノード (この場合は、xs:string アイテム)により与えられます。要素のみが型キャストノードを持つことができることに注意してください。属性は、常に、「文字列」型として処理されます。
- 名前別に子要素をグループ化します。例に関しては次を参照してください[例：名前別にノードをグループ化またはフィルターする](#)。

「動的なノード名」モード内で作業することのできるノード型は、下で説明されています。

element()

ターゲットコンポーネントと比較すると、このノードはソースコンポーネント内で異なる振る舞いをします。ソースコンポーネント内で、ノードの子要素をシーケンスとして提供します。図 3element()は、全てのproduct の子要素のリスト(シーケンス)を与えています。例えば、次のXML から作成されたシーケンスは、3つのアイテムを含みます (これは、product 3つのつの子要素が存在するからです)：

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

シーケンス内の各アイテムの実際の名前 と型は、node-name() ノード と型キャストノードによりそれぞれ与えられています 上記を理解するには、ソースからのデータXML を ターゲットXML に次のように変換する必要があると仮定してください：

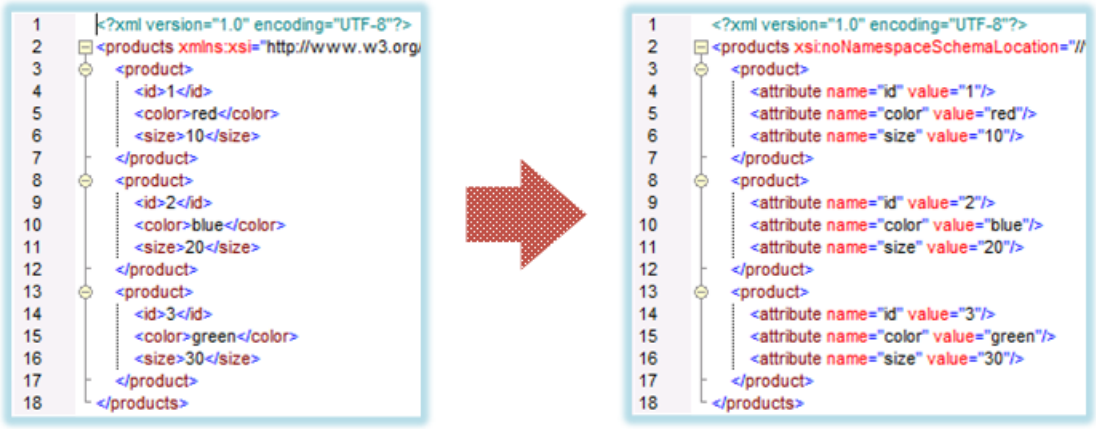


図 6 シーケンス内の各アイテムの実際の名前と型は、node-name() ノードと型キャストノードによりそれぞれ与えられています 上記を理解するには、ソースからのデータXML をターゲットXML に次のように変換する必要があると仮定してください:

マッピングの目的を達成するマッピングは以下のようになりますマッピング:

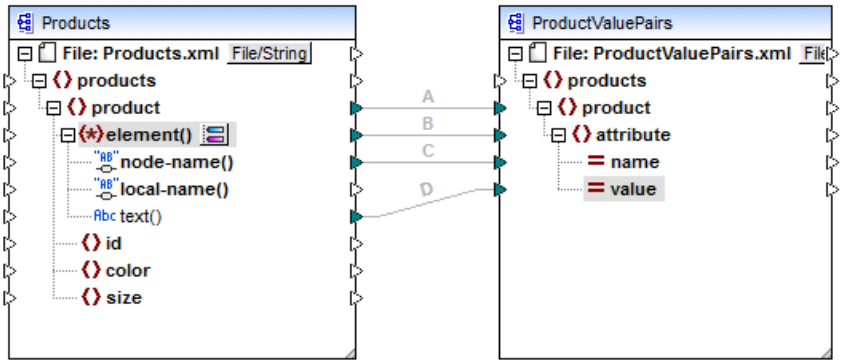


図 7 (MapForce 内で)XML 要素名を属性値にマッピングする

node-name() と text() は、シーケンス内の各アイテムの値と実際の名前を与えますが、element() のここでの役割は、product の子要素のシーケンスを提供します。このマッピングは、チュートリアルのサンプルが付随しており、以下で更に詳しく説明されています: 例:要素名を属性値にマップする。

ターゲットコンポーネント内では、element() は、ソース内の各アイテムのマッピングの基本ルールの例外であるとおり自身では何も作成しませんが、1つのターゲットアイテムを作成します。実際の要素は、(node-name()の値を使用して)型キャストノードと(自身の名前を使用して)名前テストノードにより作成されます。

attribute()

図 4内で表示されているように、このアイテムは、マッピングのランタイムでノードの全ての属性へのアクセスを有効化します。ソースコンポーネント内で、接続されたソースノードの属性をシーケンスとして提供します。例えば、次のXML 内で、シーケンスは、(product には2つの属性が存在するため)2つのアイテムを含みます:

```
<product id="1" color="red" />
```

attribute() ノードは、シーケンス内の各属性の値のみを文字列の型として与えます。各属性の名前はnode-

name() ノードにより与えられます。

ターゲットコンポーネント内では、このノードは、接続されたシーケンスを処理し、シーケンス内の各アイテムのために属性の値を作成します。属性の名前は、node-name() により与えられます。例えば、ソースからのデータXML をターゲット XML を以下のように処理すると想定します：

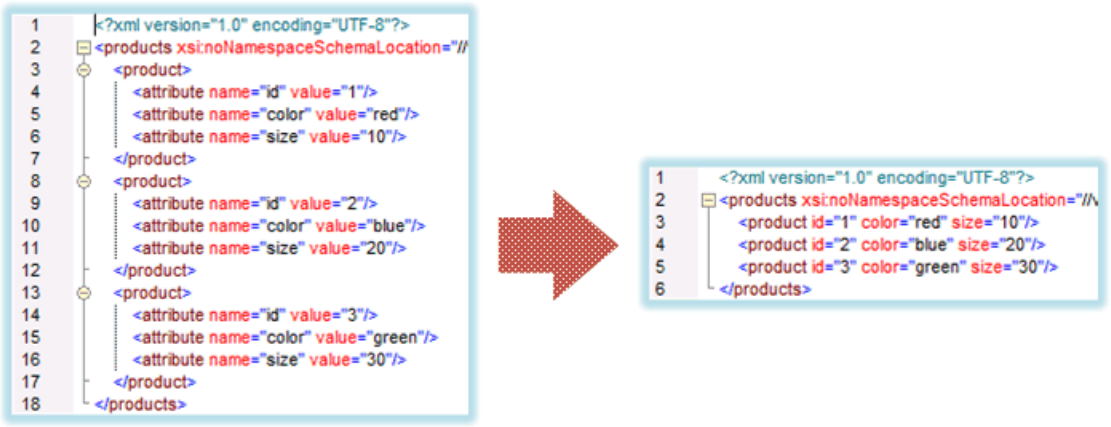


図 8　属性の値を属性名にマッピングする（必須）

この目的を達成するマッピングは以下のようになります：

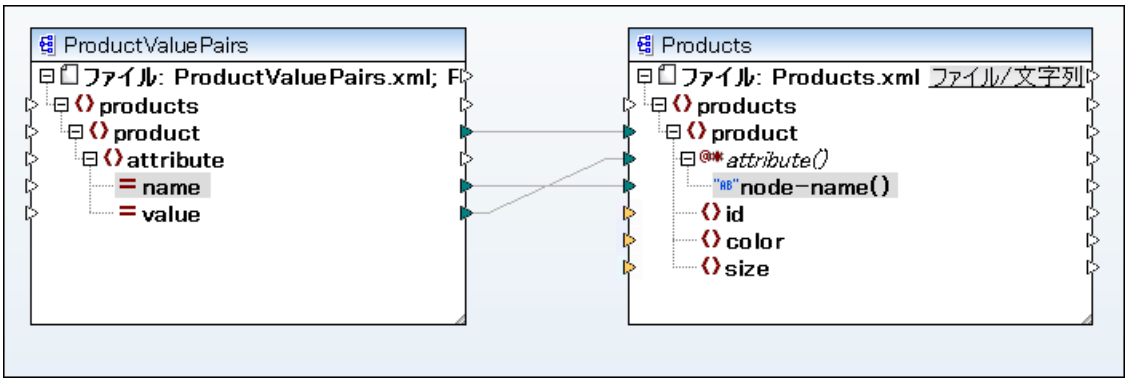


図 9　(MapForce 内で属性の値を属性名にマッピングする

メモ： この変換は、ノードの属性へのアクセスを有効化せずに行うことができます。ここでは、どのようにattribute()がターゲットコンポーネント内で動作するかを表示しています。

このマッピングを再作成する場合は、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内の**ConvertProducts.mfd** マッピングと同じXML コンポーネントを使用しています。マッピングとして使用します。唯一の違いは、ターゲットがソースになり、ソースがターゲットになることのみです。ソースコンポーネントのための入力データには、属性の値を実際に含むXML インスタンスが必要になります。例：

```
<?xml version="1.0" encoding="UTF-8"?>
<products>
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="big"/>
   </product>
</products>
```

上記のコードリスティングでは、名前空間とスキーマ宣言が、ここでは簡単にするために省略されていることに注意してください。

node-name()

ソースコンポーネント内では、node-name() は、element()の子要素の名前、または、attribute()の要素の名前をそれぞれ提供します。デフォルトでは、提供される名前は、xs:QName の型です。名前を文字列として取得するには、 local-name() ノードを(図 3を参照してください)、または、関数 QName-as-string を使用します。

ターゲットコンポーネント内ではnode-name() は、element() またはattribute()内に含まれる各要素または属性の名前を書き込みます。

local-name()

このノードは、node-name() と同様の作動をしますが、異なる点は、xs:QName の代わりにxs:string が使用されていることです。

型キャストノード

ソースコンポーネント内では、型キャストノードは、element() 内の各子要素の値を提供しますこのノードの名前と構造は、「動的に名前がつけられた子の設定」ダイアログボックス (図 2)から選択された型により異なります。

ノードの型を変更する場合は、 **「選択の変更」**( ) ボタンを使用して、スキーマワイルドカード (xs:any) を含む、使用できる型から希望する型を選択します。詳細に関しては、次を参照してください:特定の型のノードにアクセスする。

ターゲットコンポーネント内で、型キャストノードは、element() 内に含まれる各子要素の値特有のデータ型として書き込みます。希望するデータ型は **「選択の変更」**( ) ボタンをクリックすることにより選択することができます

名前テストノード

ソースコンポーネント内では、名前テストノードは、ソースインスタンスから名前別に子要素をグループ化、または、フィルターする方法が提供されています。正確な型を使用してマッピングが、インスタンスデータにアクセスしていることを保証するために子要素を名前別にフィルターする必要がある場合があります。 次を参照してください:特定の型のノードへのアクセス)。例えば、例 :名前別にノードをグループ化またはフィルターするを参照してください。see .

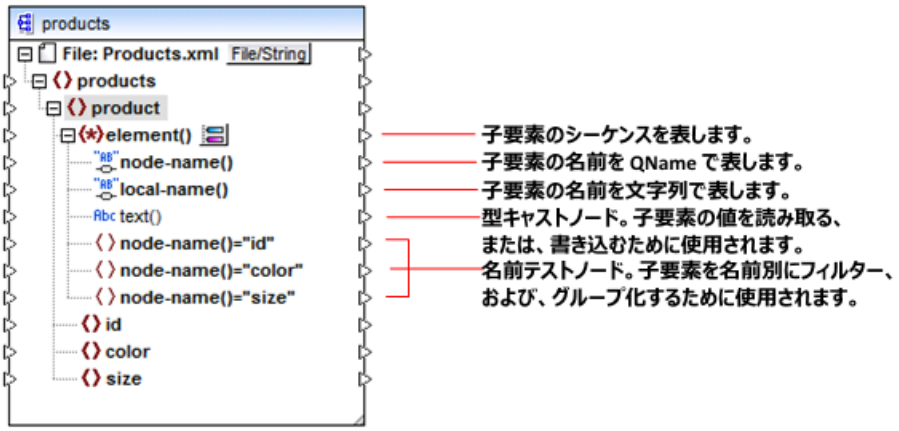
一般的には、名前テストノードは、値とサブ構造の読み取りと書き込みをする通常の要素ノードとほぼ同様の作動をします。しかしながら、動的なアクセスが有効化されていると、マッピングのセマンティクスが異なるため、制限が発生します。例えば、2つの名前テストノードを連結することはできません。

ターゲット側では、名前テストノードは、接続されているソースシーケンスにアイテムが存在するため、出力内と同じ数量の要素を作成します。これらの名前は、node-name()にマップされている値を上書きします。

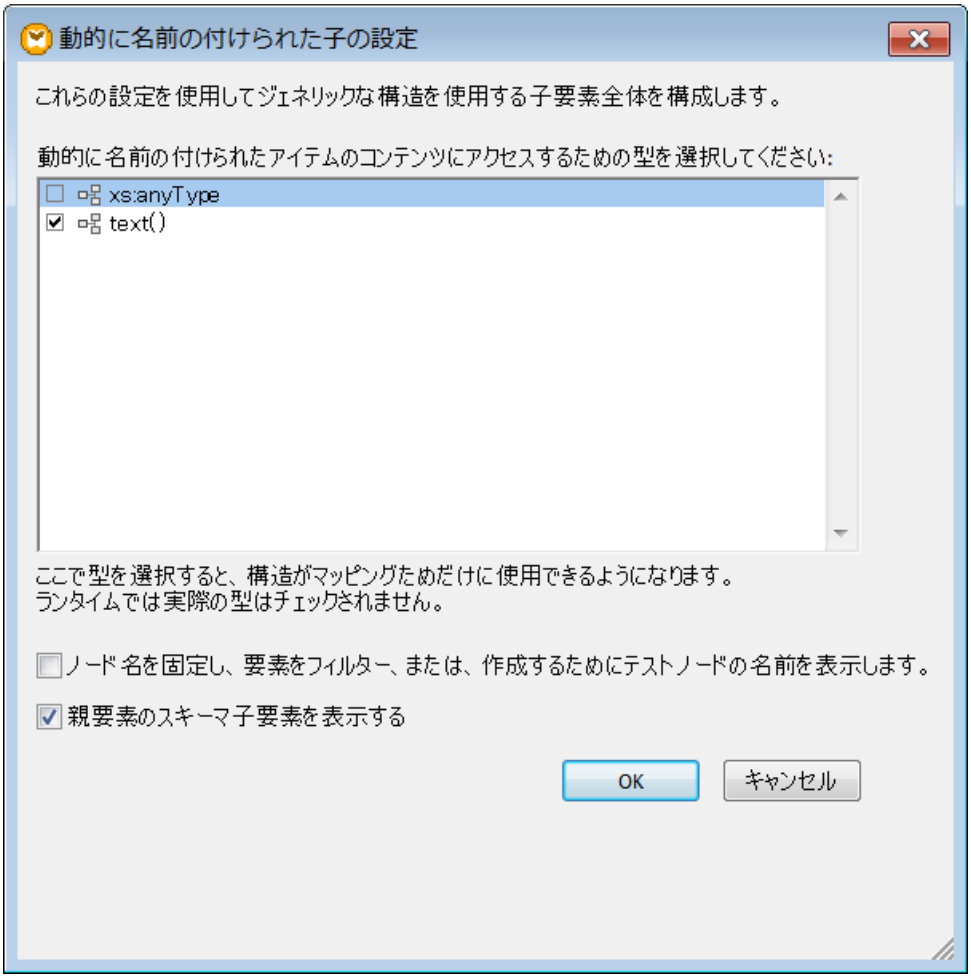
必要であれば、コンポーネントから名前テストノードを非表示にすることができます。 これを行うには、 element() ノードの横の **「選択の変更」**( ) ボタンをクリックします。そして、 「動的な名前を持つ子の設定」ダイアログボックスから、**「名前テストノードの表示」**チェックボックスのチェックを解除します。

5.14.2 特定の型のノードへのアクセス

前のセクションで説明されているとおり、[ノード名へのアクセスの取得](#) は、ノードを右クリックして **動的な名前を持つ子要素を表示する** コンテキストメニューコマンドを選択し、ノードの全ての子要素にアクセスすることができます。特定の型のノードにより値をアクセスすることはできますが、マッピングのランタイムでは、`node-name()` ノードから各子要素の名前にアクセスすることができます。下のイメージでは、型キャストノードは、`text()` ノードの横にあります。



子要素のデータ型は、マッピングのランタイム前にはわかりません。また、各子要素により異なる可能性があります。例えば、XML インスタンスファイル内の `product` ノードは、型 `xs:integer` の子要素 `id` と型 `xs:string` の子要素 `size` を持つ可能性があります。特定の型のノードコンテンツにアクセスするには、下に表示されるダイアログボックスがノードの子要素への動的なアクセスを有効化するたびに開かれます。`element()` ノードの横の **選択の変更** (　)ボタンをクリックすることにより、このダイアログボックスを後で開くことができます。



動的に名前がつけられた子の設定」ダイアログボックス

マッピングのランタイムに各子要素のコンテンツにアクセスするには、いくつかのオプションがあります:

1. コンテンツを文字列としてアクセスするには、上のダイアログボックスで**text()** チェックボックスを選択します。この場合、ダイアログボックスが閉じられると、`text()` ノードがコンポーネント上に作成されます。このオプションは、コンテンツが(`xs:int`、`xs:string` などの)単純型 の場合、適切です。例 :要素名を属性値にマップするで詳しく説明されています。**text()** ノードは、現在のノードの子 ノードがテキストを含むことができる場合表示されることに注意してください。
2. スキーマに許可されている特定の複合型としてコンテンツにアクセスすることができます。カスタム複合型が、選択されているのためにノードグローバルに許可されていると、上のダイアログボックスで使用することが可能になり、横のチェックボックスを選択することができます。上のイメージでは、グローバルに定義されている複合型は存在せず、この選択は使用することができません。
3. コンテンツを型としてアクセスする。これは、高度のマッピングのシナリオで役に立ちます (次を参照してください: 更に深い構成にアクセスする」を参照してください)。これを行うには、**xs:anyType** の横のチェックボックスを選択してください。

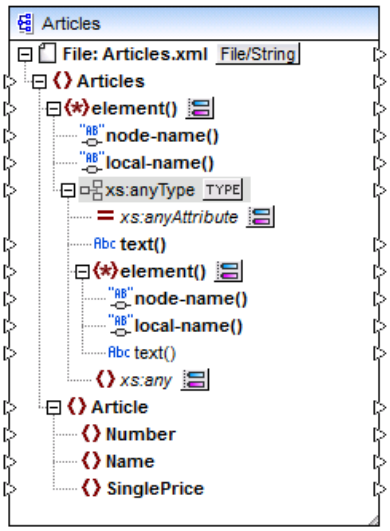
マッピング ランタイムでは、MapForce (型 キャストノードを介して)には、実際のインスタンスノード型に関する情報がないことに注意してください。ですから、マッピングは、正確な型を使用してノードコンテンツにアクセスする必要があります。例えば、ソースXML インスタンスが多種の複合型の子 ノードを持つことを希望する場合、以下を行います:
a) 型キャストノードを一致させる必要が有る複合型に設定します (上のリストの2番目のアイテムを参照してくださ

い)。
b) 一致する必要のあるインスタンスのみから読み込むためにフィルターを追加します。このテクニックは、次で説明されています：例：名前別にノードをグループ化またはフィルターする

更に深い構成にアクセスする

スキーマワイルドカードの選択により、ノードをノードの直下の子よりもスキーマ内の更に深いレベルでアクセスすることができます。高度なマッピングのシナリオではとても役に立ちます。マッピングは、XML ノードの直下の子しかアクセスすることができないため、例：要素名を属性値にマップする等の簡単なマッピングでは、このテクニックを必要としません。しかしながら、動的に更に深い構造にアクセスする必要がある場合、例えば「孫」にアクセスが必要な場合、以下の方法でアクセスすることが可能です。

1. 新規マッピングを作成する。
2. 挿入メニューから**「XML スキーマ/ファイルを挿入する」**をクリックし、XML インスタンスファイルを参照します。(この例では、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーからの**Articles.xml** ファイルです)。
3. `Articles` ノードを右クリックして、**「動的な名前を持つ子要素を表示する」**コンテキストコマンドを選択します。
4. 「動的に名前がつけられた子の設定」ダイアログボックスから**xs:anyType** を選択します。
5. `xs:anyType` ノードを右クリックして、**「動的な名前を持つ子要素を表示する」**コンテキストコマンドに対して選択します。
6. 「動的に名前がつけられた子の設定」ダイアログボックスから**text()** を選択します。



上記のコンポーネントで、2つの`element()` ノードが存在することに注目してください。2番目の`element()` ノードは、**Articles.xml** インスタンス内の`<Articles>` ノードの孫に動的なアクセスを与えます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Articles.xsd">
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
```

```
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

Articles.xml

例えば、「系」要素名（Number、Name、SinglePrice）を取得するには、2番目のelement()ノードの下のlocal-name()ノードからターゲットノードに接続を描きます。同様に、「系」要素の値（1、T-Shirt、25）を取得するには、text()ノードから接続を描きます。

この例には適用することはできませんが、実際のシナリオでは、更に深いレベルにアクセスするために、動的なノード名を後のxs:anyTypeノードに対して有効化することができます。

以下の点に注意してください：

- TYPEボタンを使用して、生成された型を現在のスキーマから選択し、異なるノード内に表示することができます。この方法は、生成されたスキーマの型から、または、スキーマの型へマップする必要がある場合のみに役に立ちます（次を参照してください：生成されたXML スキーマ型）。
- element()ノードの横の**「選択の変更」**（）ボタンは、子のトピックで説明されている「動的に名前の付けられた子の設定」ダイアログボックスを開きます。
- xs:anyAttribute属性の横の**「選択の変更」**（）ボタンにより、スキーマ内でグローバルに定義された属性を選択することができます。同様に、xs:any要素の横の**「選択の変更」**（）によりスキーマ内でグローバルに定義されている要素を選択することができます。これにより、スキーマワイルドカードへ、または、スキーマワイルドカードからのマッピングと同じように動作します（次も参照してください：ワイルドカード -xs:any / xs:anyAttribute）。このオプションを使用する場合、選択された属性または要素がスキーマに従い特定のレベルで存在できることを確認してください。

5.14.3 サンプル：要素名を属性値にマップする

この例は、XML ドキュメントから、要素名をターゲットXML ドキュメント内の属性の値にマップする方法について説明しています。この例には、付随するサンプルマッピングがあり、次のパスを使用して検索することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\ConvertProducts.mfd**。

例を理解するために、XML ファイルは製品のリストを含むと想定しましょう。各製品は次のフォーマットを有します：

```
   <product>
      <id>1</id>
      <color>red</color>
      <size>10</size>
   </product>
```

目的は、各製品の情報を名前と値ペアに変換することです。例：

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

上記のようなデータマッピングを行うには、この例でも使用されていますが、"ノード名への動的なアクセス"と呼ばれるMapForce 機能を使用しています。マッピングを実行する際、「動的」とは、(値だけではなく)ノード名も読み取ることができ、これらの名前を値として使用することを意味します。必要とするマッピングをいくつかのシンプルなステップを以下に示されるように作成することができます。

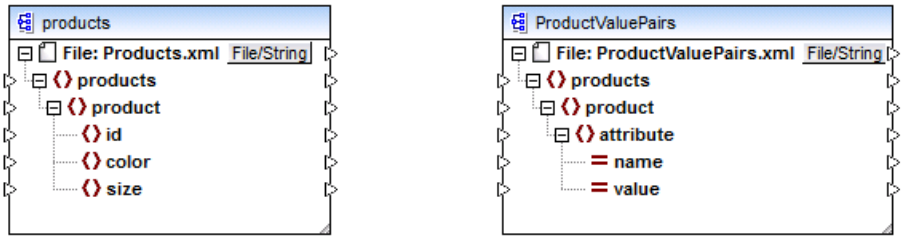
ステップ 1：ソース XML コンポーネントをマッピングに追加する

- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Products.xml** を参照します。このXML ファイルは、同じフォルダー内にある**Products.xsd** スキーマを指します。

ステップ 2：ターゲット XML コンポーネントをマッピングに追加する

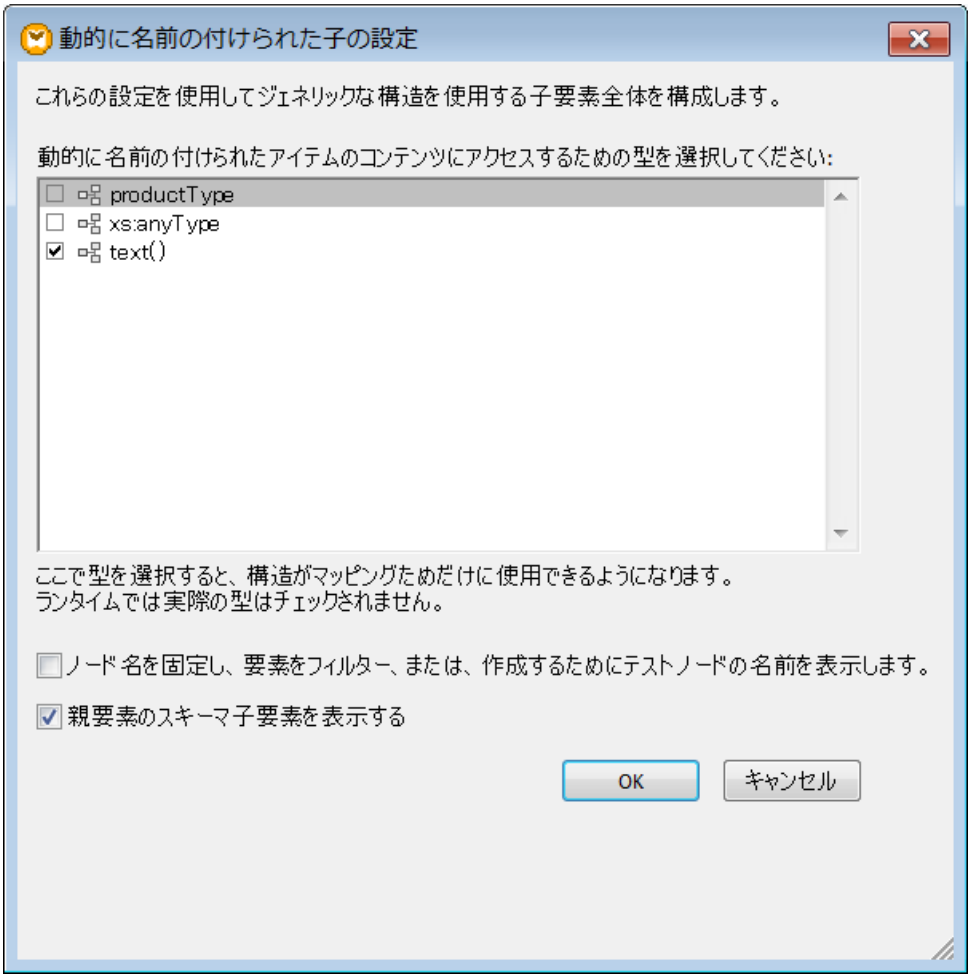
- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、次のファイルを参照します：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\ProductValuePairs.xsd**。インスタンスファイルを与えるように問われると、**「スキップ」**をクリックします。ルート要素を選択するようにプロンプトされると、ルート要素として、`products` を選択します。

この時点では、マッピングは、以下のようになります：

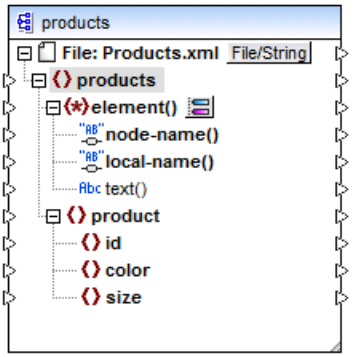


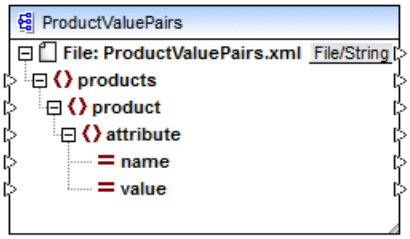
ステップ 3：子ノードへの動的なアクセスを有効化する

1. `products` ノードを右クリックして、コンテキストメニューから、**「動的な名前を持つ子要素を表示する」**を選択します。
2. 開かれるダイアログボックス内から、**text()** を型として選択します。他のオプションをそのままにします。



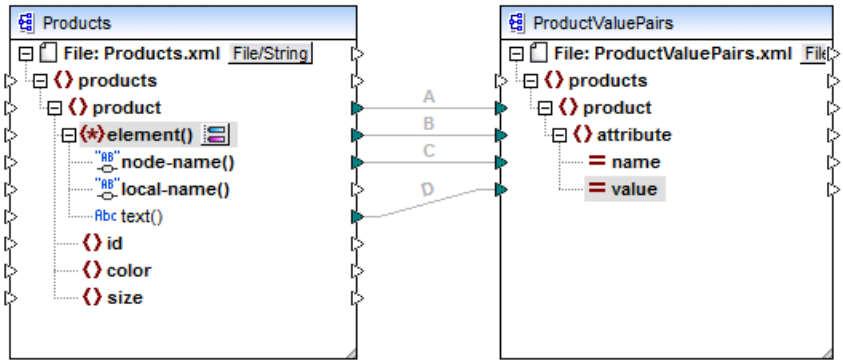
`text()` ノードがソースコンポーネントに追加されていることにご注意してください。このノードは、それぞれの子アイテムをマッピングに与えます（この場合は、「id」、「color」、「size」の値です）。





ステップ 4：マッピング接続を描く

マッピング接続 A、B、C、D 下に表示されるとおり、描きます。任意で、各接続上からをダブルクリックし、テキスト「A」、「B」「C」、および「D」をそれぞれ、詳細ボックスに入力します。



ConvertProducts.mfd

上で説明されているマッピングでは、接続 A は、ソース内の各製品のために、ターゲット内に製品を作成します。これまでは、これは ノード名を指さない標準的な MapForce 接続でした。しかしながら、接続 B は、`product` の各対応する子要素のために、新規の要素を`attribute` と呼ばれるターゲット内に作成します。

> 接続 B は、マッピング内でとても重要です。この接続の目的である`product` の子要素のシーケンスを、ソースからターゲットに運びます。実際の名前または値を運ばないため、以下として理解される必要があります：ソース**element()** にN 子要素が存在する場合、ターゲット内のアイテムの N インスタンスを作成します。この特定の場合は、ソース内の`product` には、3 つの子要素（`id`、`color` と`size`）が存在します。これは、ターゲット内の各 `product` が`attribute` という名前の3 つの子要素を持つことを意味します。
>
> この例で説明されていませんが、同じルールが、**attribute()** の子要素をマップするために使用されます。ソース **attribute()** アイテムが、N 子属性を持つ場合、ターゲット内のそのアイテムの N インスタンスを作成します。

次に、接続 C は、`product` の各子要素の実際の名前をターゲットにコピーします（文字通り、「id」、「color」、と「size」）。

最後に、接続 D は、製品の各子要素の値を文字列の型として、ターゲットにコピーします。

マッピングの出力をプレビューするには、**「出力」** タブをクリックして、生成された XML を確認してください。期待されるとおり、マッピングの意図とする目的である通り、出力は、データが名前と値のペアとして保管されている、複数の製品を含んでいます。

```
<?xml version="1.0" encoding="UTF-8"?>
<products xsi:noNamespaceSchemaLocation="ProductValuePairs.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="10"/>
   </product>
   <product>
      <attribute name="id" value="2"/>
      <attribute name="color" value="blue"/>
      <attribute name="size" value="20"/>
   </product>
   <product>
      <attribute name="id" value="3"/>
      <attribute name="color" value="green"/>
```

```
      <attribute name="size" value="30"/>
   </product>
</products>
```

生成されたマッピング出力

5.14.4 サンプル :名前別にノードをグループ化またはフィルターする

この例は、XML プロパティリストまたはXML プロパティリストからキー値ペアを読み取りCSV ファイルへ書き込むマッピングをデザインする方法を説明します。(XML プロパティリストは、OS X とiOS オブジェクトの情報をXML フォーマットで保存する方法を示します。詳細に関しては、次を参照してください :https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/PropertyLists/UnderstandXMLPlist/UnderstandXMLPlist.html。) このサンプルは、マッピングサンプルが伴い、次のパスで検索することができます :**<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial\ReadPropertyList.mfd**。

下のコードリストはソースXML ファイルを表しています。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM "https://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
   <dict>
      <key>First Name</key>
      <string>William</string>
      <key>Last Name</key>
      <string>Shakespeare</string>
      <key>Birthdate</key>
      <integer>1564</integer>
      <key>Profession</key>
      <string>Playwright</string>
      <key>Lines</key>
      <array>
         <string>It is a tale told by an idiot,</string>
         <string>Full of sound and fury, signifying nothing.</string>
      </array>
   </dict>
</plist>
```

マッピングの目的は、プロパティリストファイル内の<dict> ノードの下で検出された特定のキー値ペアからCSV ファイル内に新しいラインを作成することです。具体的には、マッピングは**<key>** - **<string>** ペアのみをフィルターしなくてはなりません。他のキー値ペア (例えば、**<key>** - **<integer>**) は、無視されます。CSV ファイル内で、リストは、プロパティの値からコンマで区切られているプロパティ名を保存しなくてはなりません。すなわち、出力は以下のようになります :

```
First Name,William
Last Name,Shakespeare
Profession,Playwright
```

この目的を達成するには、マッピングは、dict ノードのすべての子ノードへの「動的」なアクセスを使用します。第2に、XML ファイルから取得されたマッピングはキー値ペアをグループ化するためにgroup-starting-with 関数を使用します。最後に、マッピングは、フィルターを使用してノード名が「文字列」であるノードをフィルターします。

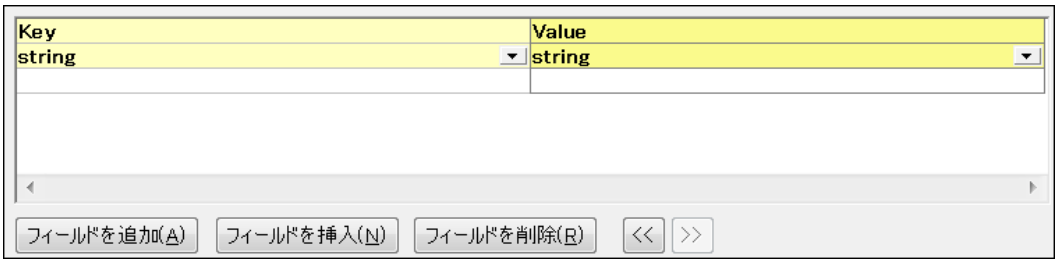
下のステップは必要とされるマッピングの作成方法を示しています。

ステップ 1：ソース XML コンポーネントをマッピングに追加する

- **「挿入」**メニューから**「XML スキーマ/ファイル」**をクリックします。次のファイルを参照します：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\plist.xml**。この XML ファイルは、同じフォルダー内の**plist.dtd** スキーマを指します。

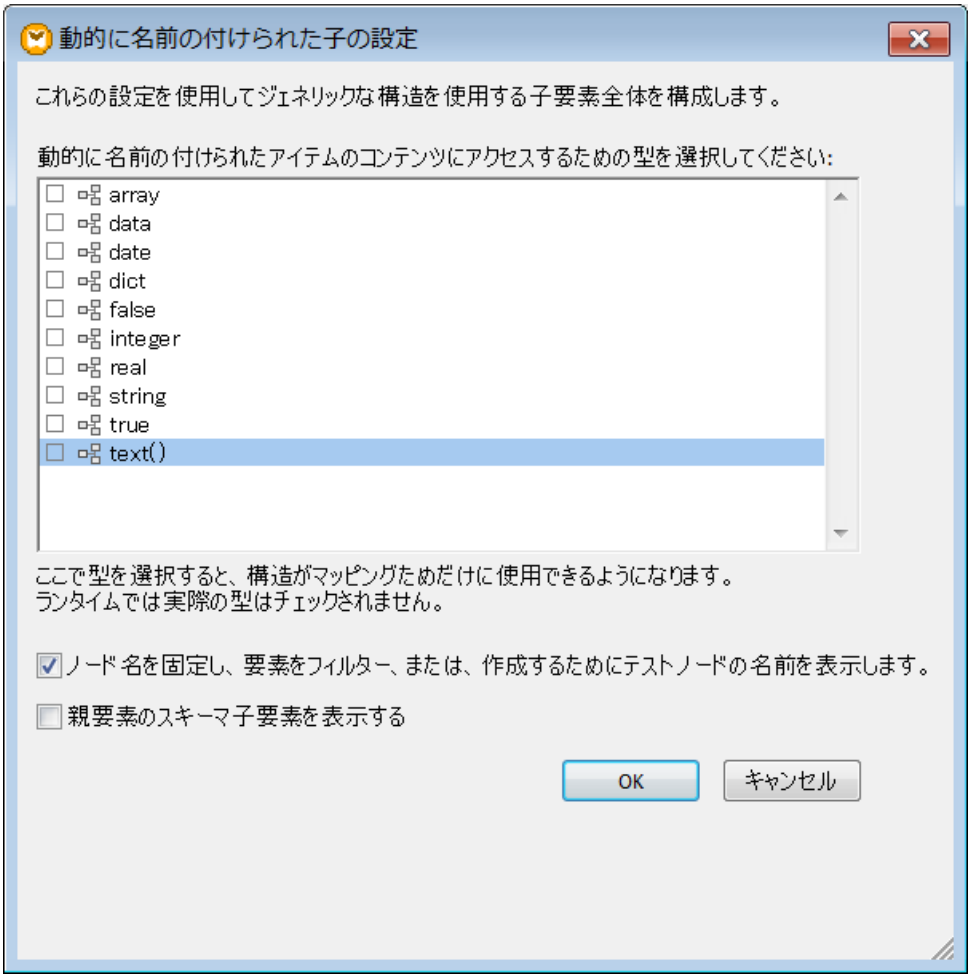
ステップ 2：マッピングにターゲット CSV コンポーネントを追加する

1. **「挿入」**メニューから**「テキストファイル」**をクリックします。プロンプトされると、**「標準 CSV のための単純処理を使用する..」**オプションを選択します。
2. **「フィールドの追加」**をクリックして、CSV フィールドをコンポーネントに追加します。ト
3. 各フィールド名をダブルクリックして、最初のフィールド名として「Key」を入力し、「Value」を2番目のフィールドの名前として入力します。「Key」フィールドは、プロパティ名を保管し、「Value」フィールドはプロパティの値を保管します。CSV コンポーネントの詳細に関しては、次を参照してください：CSV とテキストファイル.

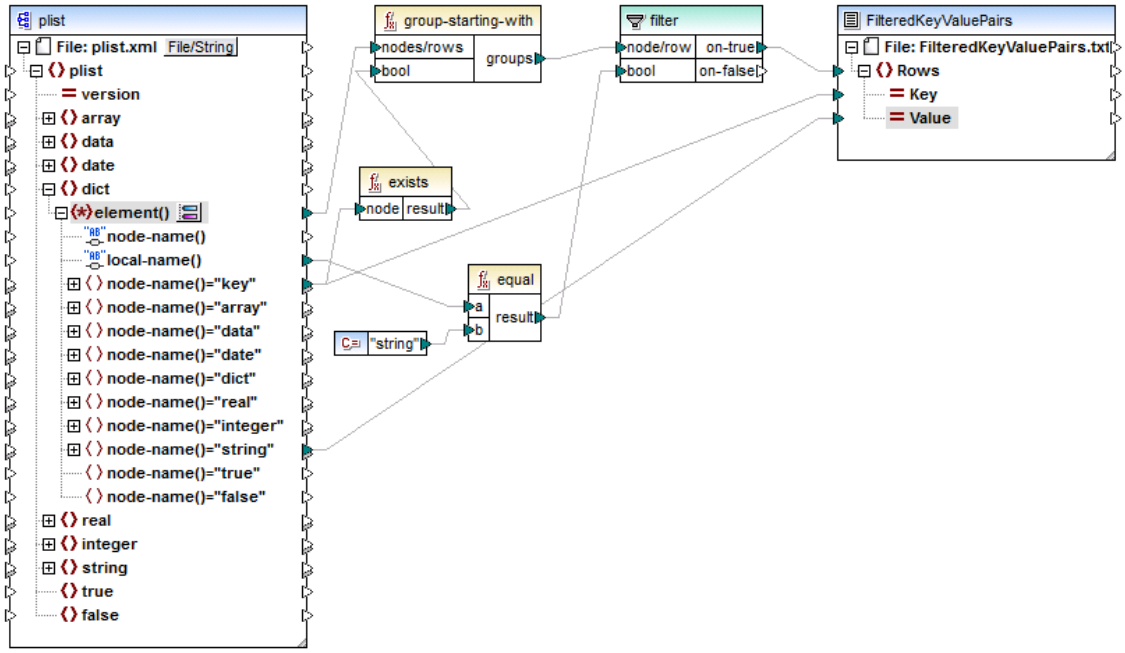


ステップ 3：フィルター と関数を追加する

1. **equal**, **exists** と**group-starting-with** 関数をライブラリウィンドウからドラッグしマッピングにドロップします。関数に関する一般的情報は次を参照してください：関数との作業.
2. フィルターを追加するには、**「挿入」**メニューから**「Filter: Nords/Rows」**をクリックします。フィルター に関する一般的情報は次を参照してください：フィルターと条件。
3. **「挿入」**メニューから**「定数」**をクリックし、テキスト「string」を入力します。
4. ソースコンポーネント内で、`dict` ノードを右クリックし、コンテキストメニューから**「動的な名前を持つ子要素を表示する」**を選択します。「動的に名前がつけられた子の設定」ダイアログボックスでは、**「ノード名を固定し、要素をフィルター、または、作成するために名前テストノードを表示する」**のチェックボックスが選択されていることを確認してください。



5. 下に表示されるように接続を描きます。



ReadPropertyList.mfd

マッピングの説明

ソースコンポーネント上の要素 ( ) アイテムは、`dict` ノードの全ての子を**`group-starting-with`** 関数へのシーケンスとして提供します。**`group-starting-with`** 関数は、名前 `key` を持つノードが発生すると新しいグループを作成します。 **`exists`** 関数は、この条件をチェックして、結果をブール値 true/false としてグループ関数に返します。

各グループのために、フィルターは、**`equal`** 関数を使用して現在のノードが「文字列」と等しいかをチェックします。名前自身は、文字列としてノードの名前を与える`local-name()` から読み込まれます。

ターゲットコンポーネントへの接続には以下の役割があります :

- フィルターの条件が、true の場合のみ、新しい行がターゲット CSV 内に作成されます。
- `Key` (プロパティ名) は、ソース内の `key` 要素の値から得られます。
- `Value` (プロパティ値) は、`string` 名前 テスト ノードから得られます。

5.14.5 マッピングのルールと戦略

MapForce は、通常、データを直感的にマップしますが、出力の結果が、多すぎるアイテム、または少なすぎるアイテムを含む場合があります。このトピックでは、上記のような問題を避けるための説明がされています。

一般的なルール

通常、ソースとターゲットアイテム間のそれぞれの接続は、以下を意味します :各ソースアイテムのために、1つのターゲットアイテムが作成されます。ソース ノードが単純コンテンツ (例えば、文字列または整数) を含み、ターゲットノードが単純コンテンツを受け入れる場合、MapForce は、コンテンツをターゲットノードにコピーし、必要であれば、データ型を変換します。

次の例外を除いて、上記は全ての接続に対して当てはまります :

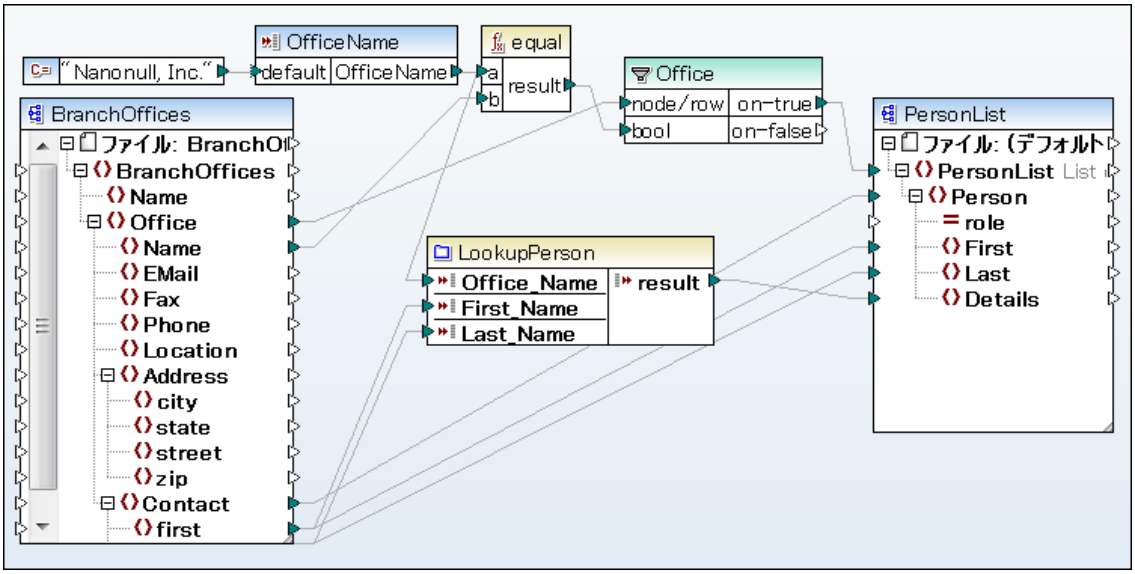
- ターゲットXML ルート要素は、通常、1度のみ作成されます。シーケンスをこの要素に接続すると、要素のコンテンツのみが繰り返され、ルートノード要素自身は繰り返されず、結果は、スキーマに対して有効でない可能性があります。ルート要素の属性はも接続されている場合、XML シリアル化は、ランタイムで失敗するため、ルート要素のシーケンスへの接続は回避されるべきです。複数の出力ファイルを作成して、達成する場合は、ファイル名を作成する関数を使用して、シーケンスを「ファイル」ノードに接続してください。
- ノードの一部は、シーケンスではなく単一の値を受け入れます。 (例えば、XML 属性、データベースフィールド、およびユーザー定義関数内の出力 コンポーネントなど)。

「コンテキスト」と「現在」のアイテム

MapForce は、スキーマ、データベース、または、EDI ファイルの構成を、コンポーネント内でマップすることのできるアイテムの階層として表示します。これらの各ノードは、インスタンスファイル、または、データベース内で、多数のインスタンスを有する場合があります。もしくは、まったくインスタンスを有さない場合もあります。

例 :**PersonListByBranchOffice.mfd** 内のソースコンポーネントを確認してください、(**Contact** の下に単一のノード**first** が存在します。**BranchOffices.xml** インスタンスファイル内には、異なる**Office** 親 ノードの下に、異なるコンテンツを有する複数の**first** ノードと**Contact** ノードが存在します。

どのソースノードが実際に選択され、データがターゲット コンポーネント アイテムにコネクタを使用してコピーされるかは(**ターゲット** ノードの)現在の**コンテキスト**により決定されます。 .



PersonListByBranchOffice.mfd

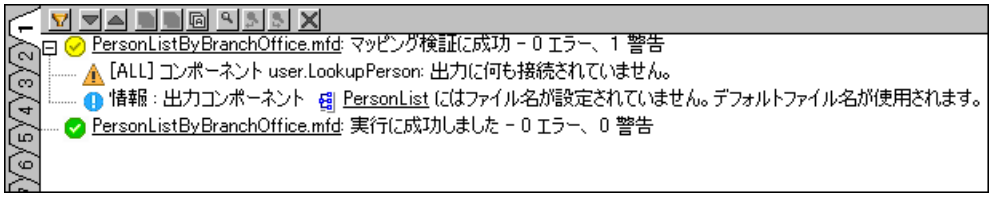
このコンテキストは、**現在のターゲットノード**により定義され、祖先に接続されます :

- 最初は、コンテキストは、ソースコンポーネントのみを含みますが特定のノードは含みません。マッピングを評価する際には、MapForce は、**ターゲット ルート** ノード (PersonList) を最初に処理します。そして、階層順に処理します。
- **ターゲット** ノードへのコネクタは、全てのソースアイテムを直接トレースすることができ、間接的に 2 つのコンポーネント間に存在する場合でもトレースすることができます。ソースアイテムと関数の結果は、このノードのためにコンテキストに追加されます。
- 各新しいターゲットノードに対して、最初は親ノードコンテキストの全てのアイテムを含む新しいコンテキストが確立されます。ターゲット兄弟ノードは、ですから、互いから独立していますが、親ノードの全てのソースデータにアクセスすることができます。

サンプルマッピング (**PersonListByBranchOffice.mfd**) に適用 :

- **Office** からフィルター (Office) を介した**PersonList** への接続は、ターゲットドキュメント全体のためのコンテキストとして**単一の**オフィスをコンテキストとします。 (これは、PersonList がターゲットコンポーネントのルート要素のためです)。オフィスの名前は、デフォルトで「Nanonull, Inc.」を含む入力コンポーネントにより与えられます。
- ルート要素 PersonList の**子孫**へのすべての接続 データは、選択された単一のオフィスがコンテキスト内に存在するため、自動的にフィルターの条件により影響されます。
- **Contact** から**Person** への接続は、1つのターゲット Person をソース XML (一般ルール)の**各** Contact アイテムのために作成します。
- **first** から**First** への接続は、現在の Contact の最初の名前を選択し、ターゲットアイテム First に書き込みます。

Contact から**Person** へのコネクタが作成されない場合、複数の First, Last、および Detail ノードを持つ1人の Person が作成されます。このような場合、MapForce は、警告を表示し、次のような問題を解決する提案をします：「Contact と Person を接続して解決することができます」:



シーケンス

MapForce は、スキーマ、データベース、または、EDI ファイルの構造を、コンポーネント内のマップすることが可能な階層として表示します。

(ターゲット)コンテキストにより、ソースコンポーネントの各マップすることのできるアイテムは以下を表します：

- 割り当てられた入力 ファイル**単一のインスタンス** ノード (または、データベース)
- 入力 ファイルのゼロから**複数の インスタンス** ノードのシーケンス (または、データベース)

シーケンスが**ターゲット** ノードに接続されている場合、ソースノードの数と同数のターゲットノードを作成するためのループが作成されます。

フィルターがシーケンスとターゲットノードの間に存在する場合、ブール値の条件は、シーケンス内の各アイテムなどの各入力ノードのためにチェックされます。具体的には、true を評価する各シーケンス内に少なくとも1つのブール値が存在するかがチェックされます。優先 コンテキスト 設定は、評価の順序を影響します。以下を参照してください。

上記の通り、フィルターの条件は自動的に全ての子孫ノードに適用されます。

メモ: ソーススキーマが特定のノードが1度のみ発生することを指定する場合、MapForce は、ループを削除し、存在が既知の場合、最初のアイテムのみを取る場合があります。この機能は、ソースコンポーネント内の設定 ダイアログボックス (「min/maxOccurs をベースにした入力処理のオプティマイゼーション」チェックボックス)内で無効化することができます。

関数 入力 (または、ノーマルな 非 シーケンス関数)は、ターゲットノードと同様に作動します。シーケンスがこのような入力に接続されている場合、シーケンス内のアイテムと**同数**作成されるようにループが関数の呼び出しの周りに作成されます。

シーケンスが**1つ以上**のそのような関数入力に接続されている場合、MapForce は、全ての入力の**デカルト積**を処理するネストされたループを作成します。通常これは、理想的ではないため、複数の (親または他のコンポーネントから単一の現在のアイテムにバウンドされている全ての他のパラメータなどの)アイテムを持つ単一のシーケンスのみを関数に接続してください。

メモ: 空のシーケンスがこのような関数 (例、concat) に接続されている場合、出力 ノードを生成しない、空のシーケンスが結果として出力されます。入力データが存在しないために、ターゲット 出力に結果がない場合、「substitute-

missing」関数 を使用して、値を代替することができます。

シーケンス入力 を持つ関数のみが入力シーケンスが**空**の場合、結果を生成することのできる関数です:

- **exists**, **not-exists** と**substitute-missing** (また、最初の3つのエイリアスである**is-not-null**, **is-null** と**substitute-null**)
- 集計 関数 (**sum**、**count** など)
- シーケンスを受け入れる通常のユーザー定義 関数 (例、非インライン関数)

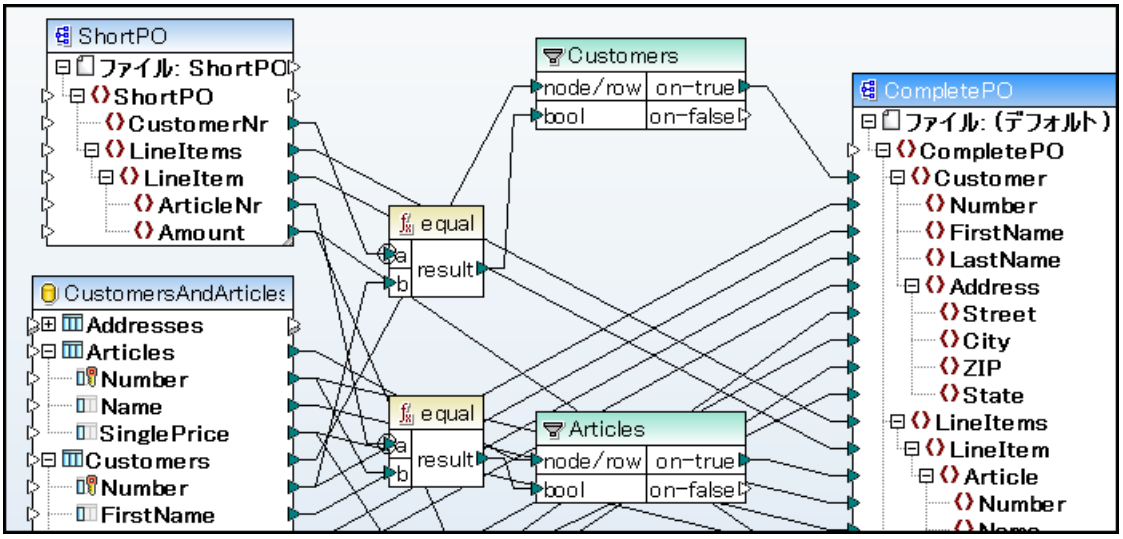
このような関数のシーケンス入力は、常に、祖先のコンテキスト内の現在のターゲットノードとは別に評価されます。コンポーネント、または、SQL-Where をフィルターする 関数は、他の接続に影響を与えません。

優先 コンテキスト

通常、関数 パラメータは、上から下に評価されますが、**優先 コンテキスト** 設定を使用して1つのパラメーターが他のパラメーターよりも先に評価されるように設定することができます。

filter 条件のブール値入力に接続されている関数では、優先 コンテキストは、比較関数自身のみではなく、フィルターの評価にも影響を与えます。このため、2つのソースシーケンスを組み合わせることができます (次を参照 :CompletePO.mfd、CustomerNo および Number)。

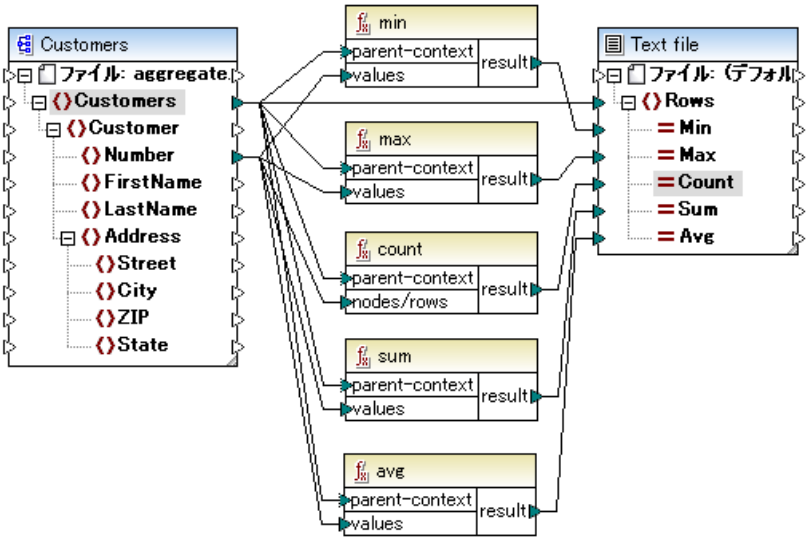
次も参照してください:優先コンテキストノード/アイテム



コンテキストのオーバーライド

集計 関数 の一部には、任意 「親コンテキスト」入力があります。

この入力が接続されていない場合、この関数は効果がなく、関数は、(ターゲットノードの親のコンテキスト内で)通常のコンテキストでシーケンス入力のために評価されます。



parent-context 入力がソース ノードに接続されている場合、関数は、**各** 「parent-context」 ノードに対して評価され、個別の結果を各発生のために作成します。次も参照してください：マッピングコンテキストのオーバーライド。

同じ ソースコンポーネントの複数のノードをコンテキストに移動させる
これは、特定のケースで必要とされ、中間関数を使用して移動することができます。

5.14.5.1 マッピングコンポーネントの処理順序の変更

MapForce は、複数のターゲットコンポーネントを有するマッピングをサポートします。各ターゲットコンポーネントには、特定のコンポーネントのためのマッピングの結果をプレビューすることのできるプレビューボタンがあります。

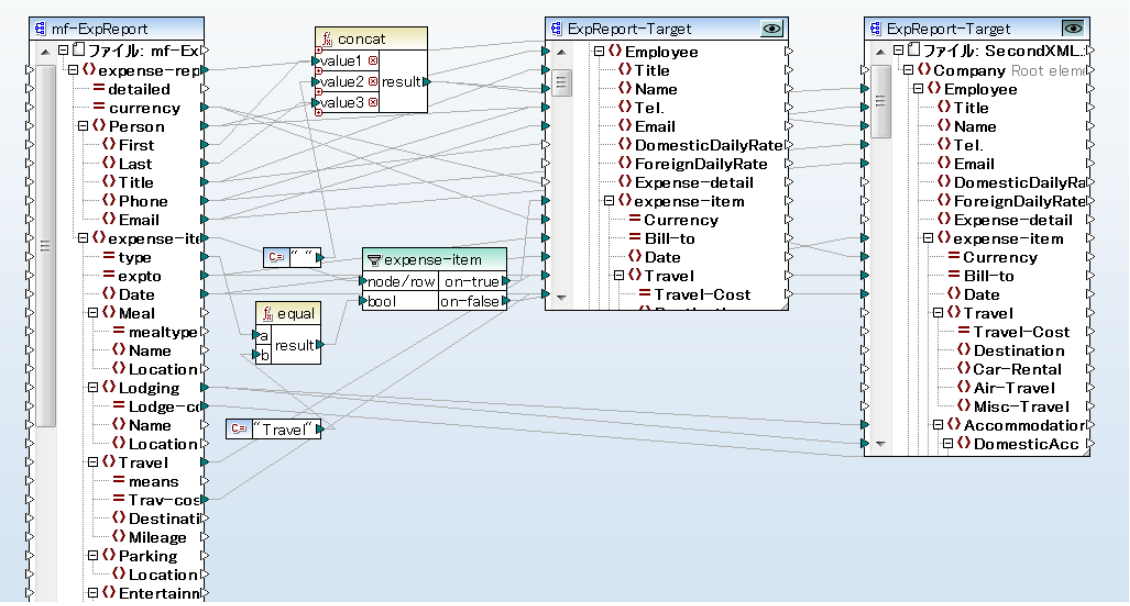
マッピングがコマンドライン、または、生成されたコードから実行されると、現在のアクティブなレビューに関わりなく、フルマッピングが実行され、各ターゲットコンポーネントのための出力が生成されます。

ターゲットコンポーネントが処理される順序は、マッピングウィンドウ内のターゲットコンポーネントの位置を変更することにより直接影響することができます。コンポーネントの**位置** は、左上と同じように定義されています。

ターゲットコンポーネントは、スクリーン上のY-X 位置に従い、上から下、左から右に処理されます。

- 2つのコンポーネントが、同じ垂直方向 位置を持つ場合、左端のコンポーネントが優先されます。
- 2つのコンポーネントが、同じ水平方向 位置を持つ場合、一番高い位置のコンポーネントが優先されます。
- コンポーネントが、完全に同じ位置を持つシナリオは少ないですが、定義されている順序を保証する、変更不可能な、ユニークな内部コンポーネントID が自動的に使用される場合はあります。

下のスクリーンショットは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内のチュートリアルサンプル **Tut-ExpReport-multi.mfd** を表しています。両方の ターゲットコンポーネント (ExpReport-Target) は、同じ**垂直方向** 位置を有し、プレビューボタンが右のターゲットコンポーネント上でアクティブ化されています。



Tut-ExpReport-multi.mfd (MapForce Enterprise Edition)

XSLT2 を選択して、コードを生成した場合：

- 左端のターゲットコンポーネントが最初に処理され、**ExpReport.xml** ファイルを生成します。
- 右のコンポーネントが次に処理され、**SecondXML.xml** ファイルを生成します。

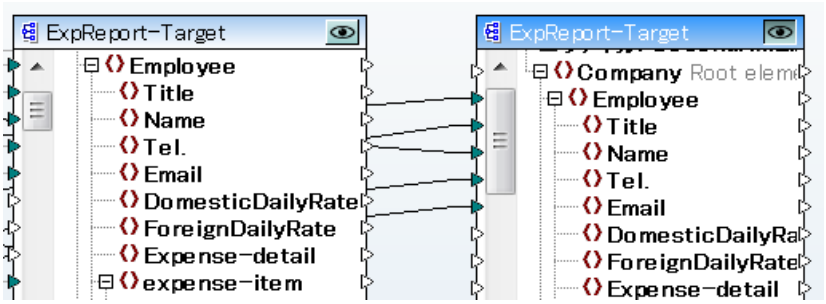
DoTransform.bat ファイル (指定された出力 フォルダー内で)開いて、生成されるシーケンス出力ファイルを確認することができます。**ExpReport-Target.xml** が、バッチファイルにより生成される最初の出力で、**SecondXML.xml** が2番目の生成される出力です。

```
@echo off

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

処理シーケンスのマッピングの変更：

1. 左のターゲットコンポーネントをクリックして、右の1つ下に移動します。



2. コードを再生成して、**DoTransform.bat** ファイルを確認します。

```
@echo off

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

SecondXML.xml は、バッチファイルにより最初に生成される出力で、**ExpReport-Target.xml** は2番目に生成される出力です。

チェーンされたマッピング
上記の同じ処理シーケンスがチェーンされたマッピングのために続きます。チェーンされたマッピンググループは、1つのユニットとしてみなされます。単一のチェーンされたマッピングの中間または最後のターゲットコンポーネント位置を変更しても、処理シーケンスに影響を与えません。

複数の「チェーン」または、複数のターゲットコンポーネントがマッピング内に存在する場合、各グループの**最後の**ターゲットコンポーネントの位置が何か最初に処理されるかを決定します。

- 最後2つのターゲットコンポーネントが同じ垂直方向位置を持つ場合、左端のコンポーネントが優先されます。
- 最後2つのターゲットコンポーネントが同じ水平方向位置を持つ場合、一番高い位置にあるコンポーネントが優先されます。
- コンポーネントが、完全に同じ位置を持つシナリオは少ないですが、定義されている順序を保証する、変更不可能な、ユニークな内部コンポーネントIDが自動的に使用される場合はあります。

5.14.5.2　優先コンテキストノード/アイテム

スキーマまたはデータベース内の異なったアイテムに関数を適用する場合、MapForceは、コンテキストノードを知る必要があります。全ての他のアイテムは、これに相対して処理されます。これは優先コンテキストとしてアイテムまたはノードをアーカイブしています。

優先コンテキストは、マッピング関連しないアイテムをマッピングする際、実行の優先順位を決めるために使用されます。

マッピングは常に上から下に実行され、2つのテーブルがループまたは検索される場合は、各ループが連続して処理されます。優先コンテキストを設定せずに関連しない要素をマップする場合はMapForceは、どのループを最初に実行するか知らないため、最初に、ですから最初のテーブルまたはデータソースを選択します。

解決方法：
どのテーブル、またはソースデータが最初にループまたは検索されるか決定し、そのテーブルへのコネクタの優先コンテキストを

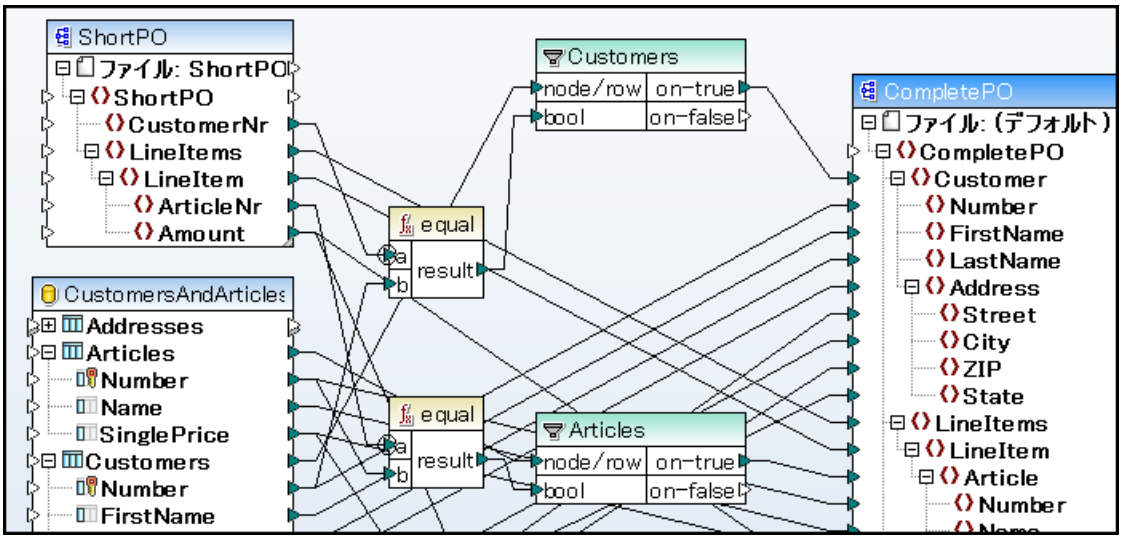
設定します。

DB_CompletePO.mfd ファイルの簡素化されたバージョンが **...\MapForceExamples** フォルダーにあり、以下に表示されています。

このサンプル内には、複数のソースコンポーネントが存在することに注意してください。**ShortPO** は、関連した XML インスタンスファイルを持つスキーマですが、**CustomersAndArticles** はデータベースです。データ CompletePO スキーマ /XML ファイルに両方からのデータがマップされます。「優先コンテキスト」アイコンは丸枠で囲まれています。

関数の**a** パラメーターを **「優先コンテキスト」**と設定すると以下の結果が発生します：

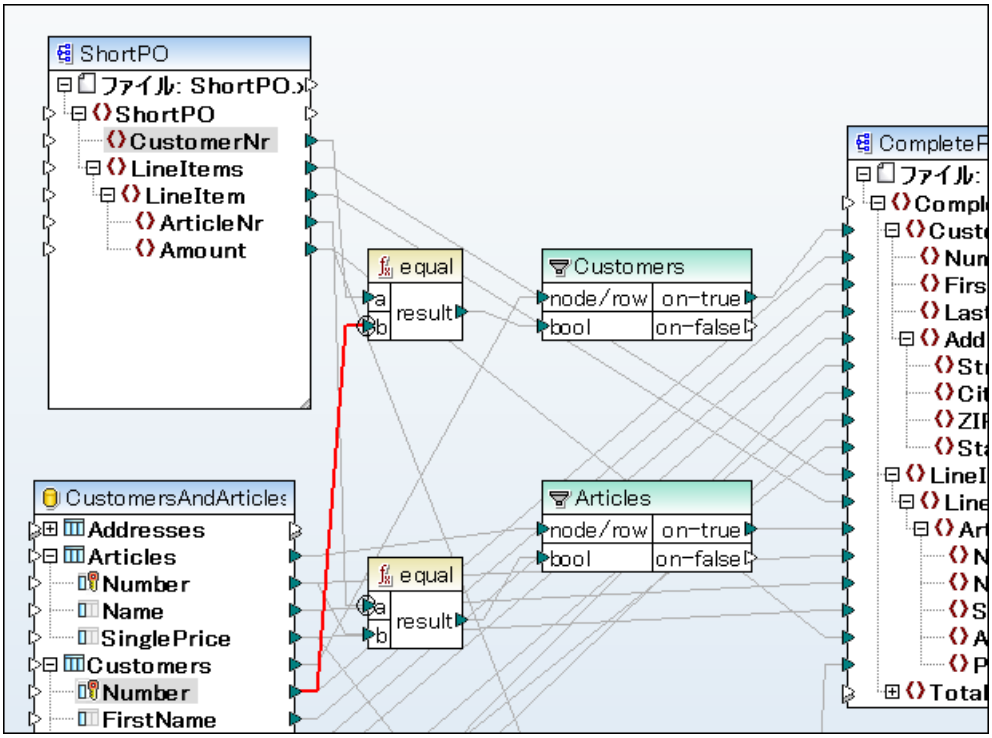
- ShortPO 内の **CustomerNr** は、データベース内のアイテム **Number** と比較されます。
- **CustomerNr** は、 **「優先コンテキスト」**と指定されており、equal 関数の **a** パラメーター内に置かれます。
- **CustomersAndArticles** データベースで**同じ**数値が (1度)検索されます。**b** パラメーターは、データベースからの Number アイテムを含みます。
- 数値が検索された場合、結果は **filter** コンポーネント (Customers) の **ブール** パラメーターにパスされます。
- **node/ row** パラメーターは、ブールパラメーター が true の場合、すなわち、同じ数値が検索された場合、**Customer** データを "on- true" にパスします。
- 残りの customer データは以下の通りパスされます :Number, FirstName、LastName アイテムはすべてのターゲットスキーマ内の対応するアイテムに接続されます。



これは、データベースが1度のみ、ShortPO により与えられたそれぞれの CustomerNr ために検索されることを意味します。

関数の**b** パラメーターを **「優先コンテキスト」** と設定すると以下の結果が発生します：

- MapForce は、データベースから最初の Number を検索し**b** パラメーター にロードします。
- **a** パラメーター内の **CustomerNr** に対してチェックします。
- 等値ではない場合、ShortPO のすべての CustomerNr を検索します。
- データベースを検索し、次の Number を **b** にロードし、a に対してチェックします。
- ShortPO 内の数値を検索するために、データベース内のすべての Number に対して反復します。

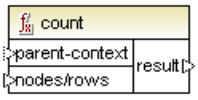


これは、各 Number のためにデータベースクエリが生成され、ShortPO のすべての CustomerNr と比較されることを意味します。

優先コンテキストとユーザー定義関数：
ユーザー定義関数がデフォルトの設定である型 "インライン"と定義された場合、優先コンテキストはユーザー定義関数のパラメーターの1つとして定義することはできません。ユーザー定義関数は、もちろん、パラメーターに設定された優先コンテキストを持つ他の正規（非インライン）ユーザー定義関数 を含んでいます。

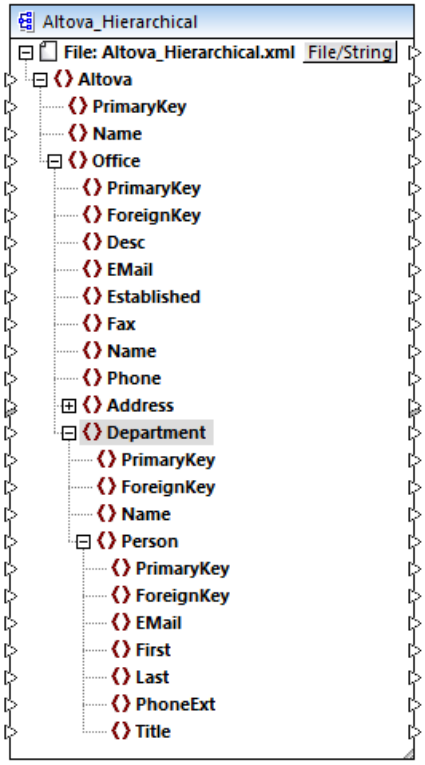
5.14.5.3　マッピングコンテキストのオーバーライド

マッピングの一部では、希望するマッピングの出力を達成するために、マッピングのコンテキストをオーバーライドする必要がある場合があります。このため、コンポーネントの一部では、必要な場合マッピングコンテキストを影響することのできる構造内の任意の`parent-context` アイテムを提供する場合があります。このようなコンポーネントのサンプルは以下のとおりです：集計関数、編集、および、ジョインコンポーネント。



任意のparent-context を使用した集計関数

マッピングコンテキストの重要性を理解するために、複数のレベルを持つネストされたノードを含むマッピング XML ファイルに追加します。 **挿入**」メニューで、**「XML スキーマ/ファイル」**をクリックし、以下のファイルを参照します：
<Documents>\Altova\MapForce2017\MapForceExamples\Altova_Hierarchical.xml。

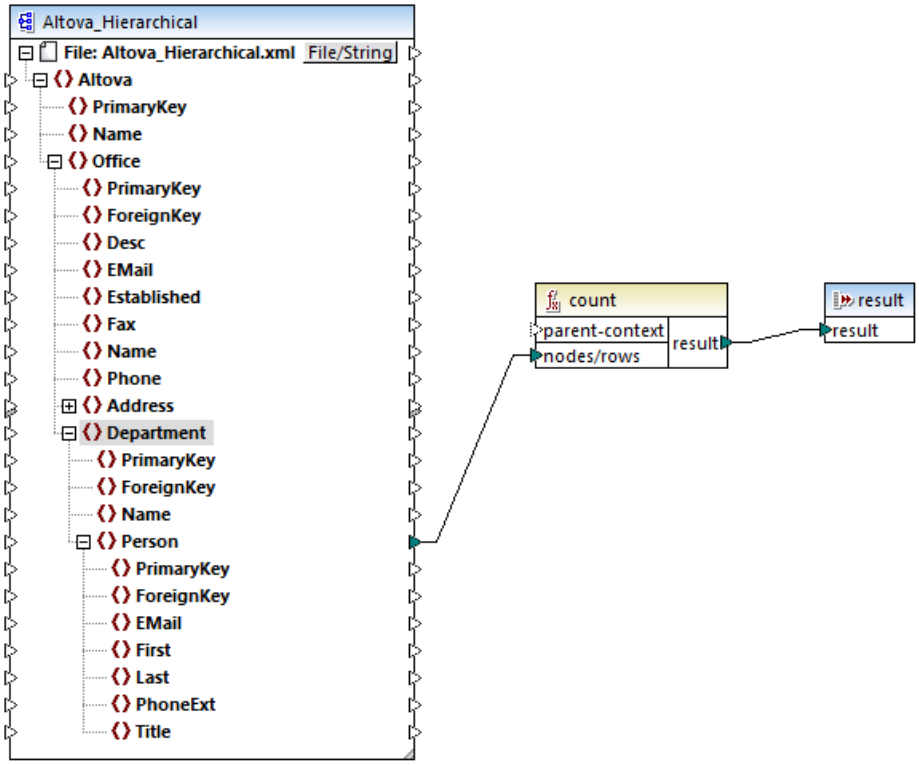


Altova_Hierarchical.xml

重要な点は、上のXML ファイル内では、`Office` 親ノードには複数の`Department` ノードが含まれており、各 `Department` には、複数の`Person` が含まれています。If you open the 実際のXML ファイルをXML エディター内で開くには、オフィスと部署の社員の所属を確認してください：

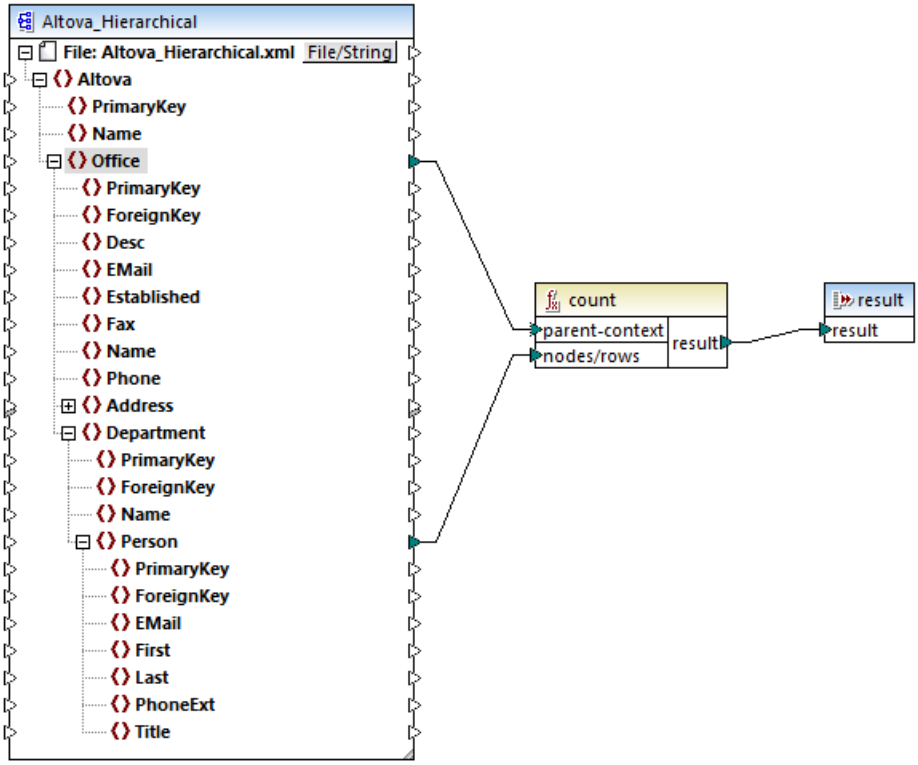
オフィス	部署	人数
Nanonull, Inc.	Administration	3
	Marketing	2
	Engineering	6
	IT & Technical Support	4
Nanonull Partners, Inc.	Administration	2
	Marketing	1
	IT & Technical Support	3

マッピングが全ての部署の全ての社員を含むと想定します。この目的を達成するためには、core | aggregate 関数から **`count`** 関数を追加し、データを以下の様にマップします：



マッピングをこの時点でプレビューすると、出力は、全ての部署内の社員の総数である21 です。**count** 関数には、まだ接続されていない任意のparent-context アイテムが含まれています。この結果、**count** 関数の親コンテキストは、ソースコンポーネントのデフォルトのルートノードです（この場合、Altova アイテム）。これは、全ての部署の全ての社員が **count** 関数の対象と考えられています。マッピングのルールと戦略で説明されているとおり、デフォルトでは、マッピングコンテキストはこのように作動します。マッピングのシナリオの大半はこれだけで十分です。

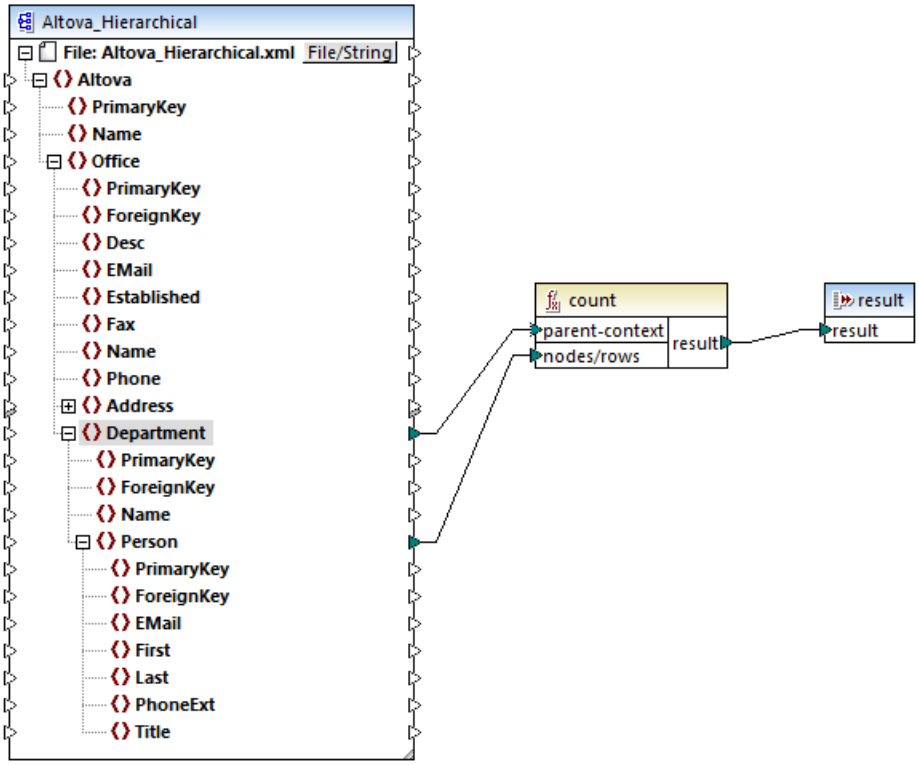
しかしながら、デフォルトの マッピングコンテキストを必要な場合はオーバーライドすることもできます。これを行うには、Department ノードからparent-context アイテムへの接続を下に表示されるように追加します。



上に示されるようにマッピングを変更することにより、マッピングに各オフィスのコンテキスト内で社員のレコードを反復するように命令します。このため、マッピングをプレビューすると、出力は、`15*` に成ります。これは最初のオフィス「Nanonull, Inc.」内の社員数です。社員ノードが（各オフィスのために1度ごと）2回計算されていることがわかります。各オフィスの社員数は、それぞれ15人と6人です。しかしながら、最初の結果のみが返されています（これは、関数が値のシーケンスを返すことができず、簡単な値のみを返すからですbecause 関数）。

* *マッピングのターゲット言語がXSLT 1.0.以上と仮定しています*

下に示されるようにマッピングコンテキストを`Department`, に変更することができます。今度は、社員のレコードが各部署のコンテキスト内で数えられます（すなわち、部署の総数に対応する 7回）。また、最初のオフィスの最初の部署内の社員数に対応する、出力が3である最初の結果のみが返されます。



このマッピングは多くがまだ発生していませんが、この時点では、マッピングの出力を`parent-context` アイテムがどのように影響するかを説明しています。この点を考慮して、変数 または、ジョインコンポーネントなどを含む他のマッピング内の `parent-context` をオーバーライドすることができます。次も参照してください:[サンプル:レコードのグループ化およびサブグループ化](#).

Chapter 6

マッピングのデバッグ

6 マッピングのデバッグ

MapForce は、MapForce 内蔵の変換言語のためのマッピングデバッガーを含みます。マッピングデバッガーは、以下の目的を達成する手助けをします：

- マッピングにより生成された値を個別のコネクタレベルで確認して分析します。
- 特定の値を作成するためのコンテキスト（ノードのセット）をマッピングでハイライトします。
- 各値をリアルタイムで処理または計算するかを確認するため、または、生成されるマッピング出力をプレビューするため、MapForce をステップバイステップで実行します。
- 目的（ブレークポイント）を、マッピングの実行に設定し、現在処理されている値を停止し、表示します。
- 現在の実行の箇所までのマッピングの実行により処理された値の履歴を確認します。

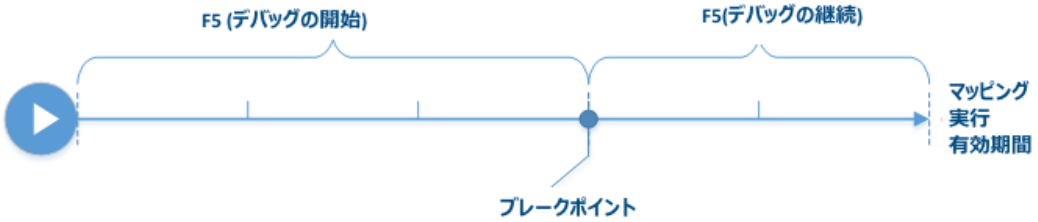
マッピングの変換言語がBUILT-IN 場合、マッピングデバッガーを使用することができます。異なる言語のためにデザインされたマッピングのデバッグを開始すると、マッピングの言語をBUILT-IN に変換するようにプロンプトされます。メニューコマンド **出力 /内蔵の実行エンジン** を選択することにより、マッピングをBUILT-IN にすることにより変更することもできます。どちらの場合でも、マッピングがBUILT-IN 言語内（例：XSLT 関数）で使用することのできないコンポーネントを含む場合BUILT-IN への変換は成功します。

MapForce デバッガーは、（MapForceを使用してコードを書く必要がないため）従来のデバッガーと異なり、プログラムコードをラインごとにトラバースしません。代わりに、MapForce により生成されたコードの結果は、デバッガーにより表示されます。具体的な方法は、デバッガーは、入力と出力コネクタを介してパスされた値をログします。ログされた値は、マッピング専用のウィンドウを介して、直接分析のために使用することができます。

以下のセクションは、マッピングデバッガーを使用する複数の方法をハイライトします。

ブレークポイントを使用するデバッグ

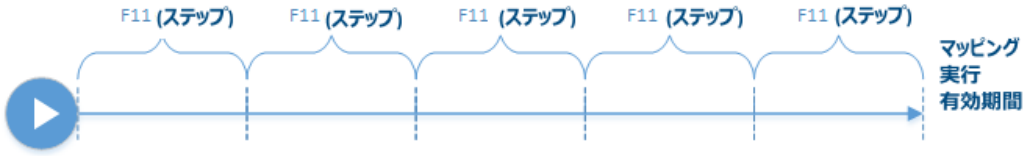
マッピングの特定の箇所で、デバッグの実行を停止する必要がある場合、従来の開発環境で行われるように、ブレークポイントを設定します。相違点は、ブレークポイントはコードのラインに追加されるのではなく、マッピングコンポーネントの入力または出力コネクタに追加されます。ブレークポイントの条件も追加することができます（これは、設定された条件が満たされると実行を停止する場合役に立ちます）。



ブレークポイントを希望するコネクタ上で定義し、最初のブレークポイントに到達するまでマッピングを実行し、また次のブレークポイントまでマッピングを継続することができます。このようにして、選択されたコネクタと関連したマッピングコンテキストと値を分析することができます。デバッガーにより与えられた **「ステップ イン」、「ステップアウト」、「ステップオーバー」**、および **最小ステップ** コマンドを使用して実行のスピードを遅くしたり速くしたりすることができます。これらのコマンドによりマッピングの一部をスキップすることができ、またその反対に、必要であればマッピングの一部を更に詳しく実行することもできます。

ステップバイステップのデバッグ

デバッグをステップバイステップで行い、マッピングコンテキストとそれぞれのステップに関連した値を分析することができます。このシナリオは前のものと類似しており、**「ステップイン」**、**「ステップアウト」**、**「ステップオーバー」**および**最小ステップ**コマンドを使用して実行のスピードを速くまたは遅くすることができます。



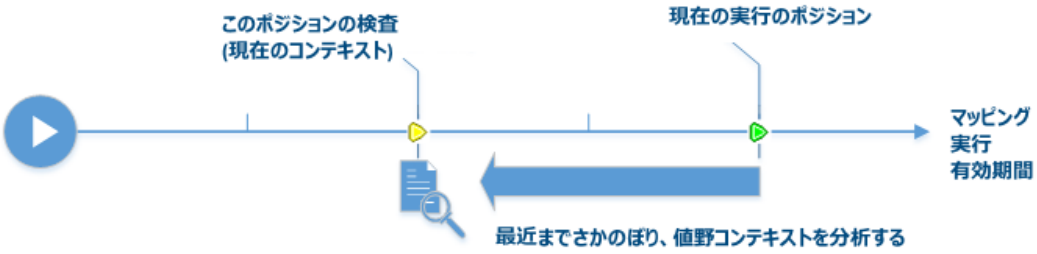
値のログを分析する

MapForce にマッピングのデバッグ中に全てのコネクタにより処理された全ての値を記憶 (履歴のトレース)するように構成することができます。履歴のトレースを保持することは、データを中心としたマッピングには、向いていないかもしれません。ですから、このオプションは、必要であれば無効化することができます。オプションが有効化されていると、各コネクタにより処理された値の現在の実行のポジションまでの、完全なログを分析することができます。値の作成の理由を理解することのできる、MapForce に特定の値に関連したマッピングを再作成することもできます。



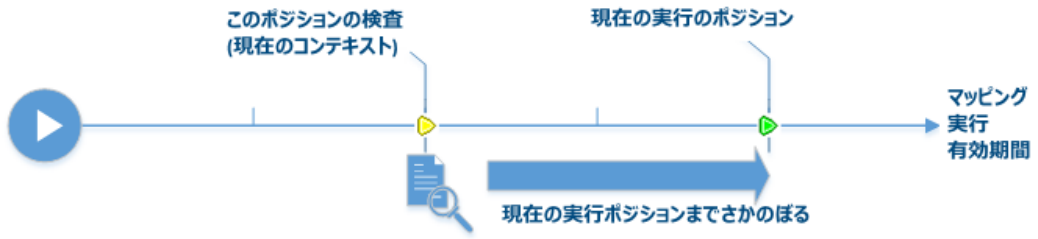
現在の実行の箇所に関連した値のコンテキストを設定する

デバッガーがマッピングで特定の実行の位置にある場合、現在の実行の箇所に対応した過去の値のコンテキストを分析することができます。(これは、少し前の時間に戻ることにも比較することができます):



コンテキストはなぜ値が計算されるかを説明します。すなわち、生成されるマッピングの特定の値について説明します。コンテキストとは、MapForce により近い過去のコンテキストとして設定することもできますが、通常現在の実行の箇所です。コンテキストが特定の値に設定されている場合、MapForce は関連するノードをマッピングで直接ハイライトし、マッピングコネクタの横にヒントを表示し、デバッガーに関連したウィンドウ内 (**値**、**コンテキスト**と**ブレークポイント**ウィンドウ) で追加情報を表示します。

現在の実行の箇所と同じではない、マッピングのコンテキストを確認した後、コンテキストを現在の場所に再設定することができます:



制限

- 現在、マッピングデバッガーは、Visual Studio のためのMapForce プラグインまたはEclipseではサポートされていません。
- MapForce がマッピングを実行する際、内部でコードを最適化します (例 :データをキャッチ、または任意の箇所で中間結果を計算する場合)。これにより、特定のコネクタ(つまりブレークポイント)はデバッグで到達できない場合があります、この場合、MapForce は通知を表示します。MapForce コード最適化は、マッピングに対するマッピング出力 (をしてこの結果、デバッガーの振る舞い)は同じ場合でも、MapForce のリリースにより異なる場合があります。
- デバッガー1つのターゲットコンポーネントのための出力をデバッグすることができます。マッピング上に複数のターゲットコンポーネントが存在する場合、デバッガーにより実行されるコンポーネントを選択する必要があります。
- 現在、データベーステーブルアクションデータベースコンポーネントの (全て挿入 " や "If の場合更新 "など)では、バッギングはサポートされません。
- ブレークポイントは次のエンティティに追加することはできません :定数、`core | position` 関数、 全てコピー」接続の子孫アイテム、 "インライン"ユーザー定義関数のパラメーター。

6.1 デバッガーの準備

デバッグの準備は、大きなシステムメモリの実行を必要とする通常大きなデータマッピングに対して必要とされます。大きな入力または出力ファイルを処理するマッピングの場合、または、大きなデータのコレクションを繰り返す場合などが当てはまります。

デバッグのスピードを向上し、メモリの使用量を減らすために、デバッグを開始する前に以下を行うことが奨励されます：

- マッピングが複合型の場合、デバッグが必要のないマッピングの一部の接続を削除または解除してください。
- マッピング大きな入力ファイルを使用する場合、小さなサイズのファイルと置き換えてください。
- **「ウルトラーズ履歴を保持」**が無効化されていることを確認してください（デバッガー設定 を参照）。

正確な出力をデバッグするために、以下を適用することができるか確認してください：

- マッピングに複数のターゲットコンポーネントがある場合、ターゲットコンポーネントを選択して、**「プレビュー」**ボタン（）をクリックします。
- マッピングがチェーンマッピング（チェーンマッピング を参照）の場合、中間コンポーネントの**「パススルー」**（）ボタンをリリースします。「パススルー」コンポーネントのデバッグは現在サポートされていません。

オプションで、デバッガーが重要なコネクタの一部を停止する場合、値を分析するこれらのコネクタへブレークポイントを追加します（ブレークポイントの追加と削除を参照）。

6.2 デバッガーコマンド

次のデバッガーコマンドにアクセスすることができます：

- **「デバッグ」**メニュー内
- キーボードのショートカット
- デバッグツールバー内

メニューコマンド	キーボードショートカット	ツールバーボタン	説明
**デバッグ	デバッグの開始**	**F5**		デバッグを開始または、ブレークポイントに到達またはマッピングが完了するまで継続します。
**デバッグ	デバッグの中止**	**Shift + F5**		デバッグを停止します。このコマンドはデバッグモードを終了して、MapForce を標準モードに切り替えます。
**デバッグ	ステップイン**	**F11**		マッピング内で単一のステップを終了するまでマッピングを実行します。マッピングでデバッガーは、シーケンスの単一アイテムを生成する独立した計算の理論的なグループです。 マッピングコンテキストにより、このコマンドは "左へ移動 /ターゲットに移動 子 /ソースの親へ移動 " と解釈することができます。
**デバッグ	ステップオーバー**	**F10**		現在のステップが終了するまで (または他のシーケンスのアイテムを終了するまで) 実行を継続します。このコマンドは、現在のステップの入力である計算をステップオーバーします。
**デバッグ	ステップアウト**	**Shift + F11**		現在のステップの結果が使用されるまで、または入力または使用の子ではないステップが実行されるまで実行を継続します。このコマンドは現在の計算をステップアウトします。 マッピングコンテキストにより、このコマンドは "右へ移動 / ターゲットへ移動 親 /ソースの子へ移動 "と解釈することができます。
**デバッグ	最小ステップ**	**Ctrl + F11**		値が生成または使用されるまで、実行を継続します。このコマンドは、ステップを細分化し、接続ごとに通常 2回停止します：ソースが値を生成した際、ターゲットが値を使用した際。MapForce は、マッピングの提示に従い値を計算する必要はなく、生成と使用イベントは順序に従わない場合があります。



デバッグツールバー

6.2.1 デバッグモードについて

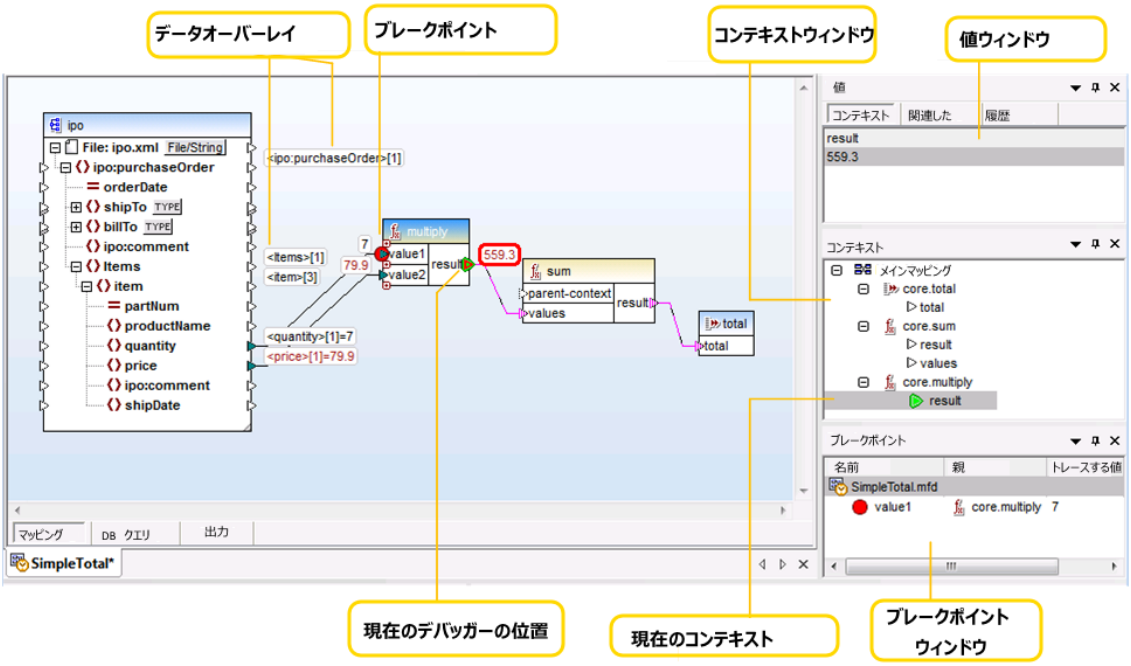
デバッグを開始すると、MapForce は、（**F5**」または「**F11**」、または「**Ctrl + F11**」を押して）マッピングをデバッグモードで実行します。

> MapForce がデバッグモードの場合、マッピングは読み取り専用です。マッピングエリアでコンポーネントを移動させることはできますが、コマンドの多くを使用することはできません。これは、マッピングの検証、デプロイ、コードの生成、マッピングのドキュメント、マッピングエリアに新規コンポーネントを追加する、または、既存のコンポーネントを再ロードするなどが含まれます。

デバッグモードにより特定の値を生成するコンテキストを分析することができます。この情報は、マッピング上および、値、コンテキスト、とブレークポイントウィンドウで直接使用することができます。デフォルトでは、これらのウィンドウは、デバッグを開始すると表示され、デバッグを中止すると隠されます。

MapForce は、「**Shift + F5**」を使用して、（または「**デバッグの中止**」ツールバーボタンをクリックすることにより）デバッグが中止されるまで、デバッグモードにあります（マッピングは読み取り専用です）。

以下のイメージは段階的にデバッグされるサンプルのマッピングを表しています（**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリからの**SimpleTotal.mfd**）（「**F11**」を押すと、先に進みます）。



デバッグモード中のMapForce 開発環境

デバッグモード中にMapForce により提供される他の情報と視覚的ヒントは以下で説明されています。

マッピングペイン

デバッグ中、マッピングペインは追加情報を表示します：

- データオーバーレイ (下を参照してください)は現在の値と関連した値をコネクタの近くに表示しています。
- (コンテキストウィンドウ内で構造と表示されている)現在のコンテキストは以下のようにハイライトされています:
  - コンテキスト内のコネクタ縞のマゼンタで表示されています ( )。
  - 曖昧なコンテキスト内のコネクタは点線のマゼンタで表示されています ( )。
- 現在の実行の場所は、緑のコネクタアイコン ( )で表示されています。

データオーバーレイ

各コネクタにより処理された値は、対応するコネクタの近くに(小さな三角形の)データオーバーレイとして表示されています。最近選択されたオーバーレイは赤い枠で表示されます。前回のステップから変更された値は、暗い赤で表示されています。単純なコンテンツを持つノードには、データオーバーレイは以下の2つの値を結合します - ノード名と値。ノード名が現在の実行ポジションの前に複数回反復されると、現在の反復のインデックスは角かっこ内の数値で表されます。

データオーバーレイは以下の振る舞いを行います:

- データオーバーレイにマウスをポイントすると、一時的に前景に常時されます。クリックすると常時前景に表示されます。また、クリックすると対応するコネクタを選択します。
- データオーバーレイはドラッグして移動することができます。
- データオーバーレイはコンポーネントが移動されると、移動します。ですから、データオーバーレイが重なって表示される場合はコンポーネント同士が近すぎるためであり、コンポーネントをマッピングエリア内で移動してスペースを与えてください。データオーバーレイもコンポーネントと共に移動します。
- データオーバーレイをクリックすると、値ウィンドウ内に値が表示されます。
- コネクタをクリックすると、データオーバーレイを選択します。

ブレークポイント

ブレークポイントは、デバッグモード中にマッピングの実行を中断するマイルストーンを指します。この用語は、他の統合された開発環境などで使用されているため、既知かもしれません。コードのラインにブレークポイントを追加する他の開発環境とは異なり、MapForce は、入力または出力コネクタに(接続の左または右側に小さな三角形で表示される)ブレークポイントを追加することができます。マッピングペインでは、ブレークポイントは赤いサークルで表示されています。定義されたブレークポイントはブレークポイントウィンドウ内に表示されます。 ブレークポイントの追加と削除を参照してください。

現在のデバッガーのポジション

緑の三角形 ( ) はデバッガーを表しています。このポジションはコンポーネントの入力または出力コネクタです。

現在処理されている値は、 **「コンテキスト」**タブの「値」ウインドウでも表示されます。

接続 および/または コネクタのセットは、縞のマゼンタで現在のマッピングコンテキストを表します。同じ情報が「コンテキスト」ウインドウ内で階層構造として表示されています (コンテキストウィンドウの使用 を参照)。

値のコンテキストを手動で設定する場合、現在のデバッガーのポジションは最も最近の実行ポジションに相対的な過去にあるポジションです。最も最近のポジションと過去のポジションの違いを明確にするために、現在のポジションのコネクタは、デバッガーインターフェイス内で以下の色別に表示されています。

	緑は "現在 "を意味し、現在の実行のポジションを示します (コネクタの現在の値を表示するを参照してください)。
	黄色は "過去 "を意味し、現在の実行のポジションに相対して、過去のコネクタを観察していることを示し

	ます。これは、コンテキストを手動で設定した後に発生する場合があります（コンテキストを値に設定するを参照してください）。

値 ウィンドウ

値ウィンドウは、マッピングにより処理された値についての情報を提供します。現在の実行のポジション、または、ユーザーが設定した特定のコンテキストでどのマッピング処理されたかについての情報が提供されます。値ウィンドウの使用を参照してください。

コンテキストウィンドウ

コンテキストウィンドウは、現在のデバッガーのポジションに関連したノードのセットと関数の階層的なビューを提供します。 コンテキストウィンドウの使用を参照してください。

ブレークポイント ウィンドウ

ブレークポイントは、MapForce が開始されてから作成されたデバッグ ブレークポイントのリストを表示します。複数のマッピングでブレークポイントを定義した場合、ブレークポイントウィンドウ内に全てのブレークポイントが表示されます。 ブレークポイントウィンドウの使用を参照してください。

6.3 ブレークポイントの追加と削除

ブレークポイントは、デバッグモード中の実行でどのマッピングをブレークするかを指定するマイルストーンです。作成されるブレークポイントは、すべてのマッピングのために グローバルに保管され、ブレークポイントウィンドウで表示されます。 ブレークポイントは明示的に削除されるか、MapForce が閉じられるまで有効です。

メモ: ブレークポイントは以下のエンティティには追加することはできません:定数、`core | position` 関数、 「全てコピー」接続の子孫アイテム、 "インライン"ユーザー定義関数のパラメーター。

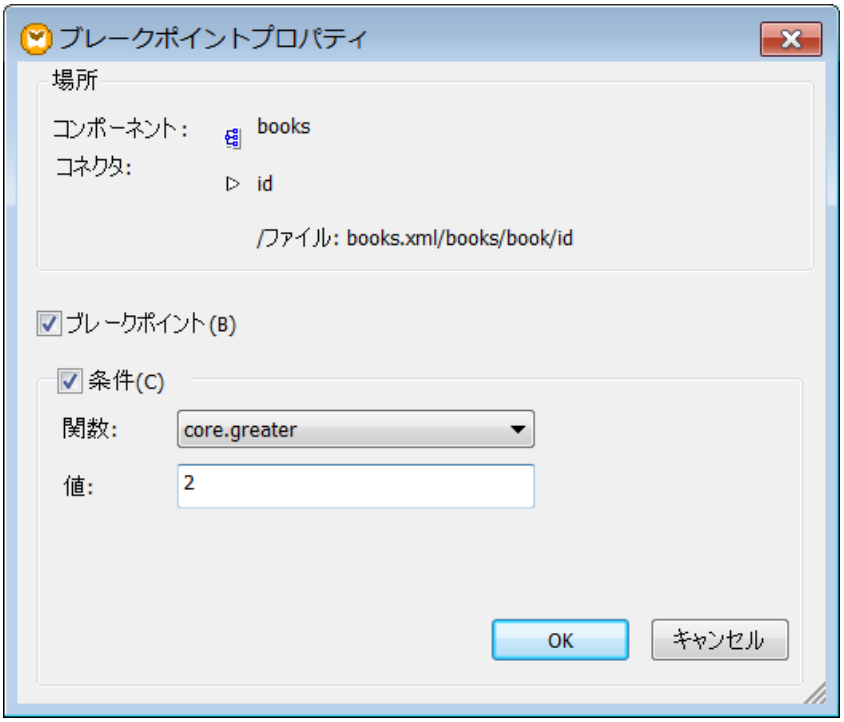
ブレークポイントは簡単または条件付きで作成することができます。簡単なブレークポイントはマッピングの実行を条件無しに停止します。条件付ブレークポインはマッピングの実行を、割り当てられた条件が満たされた場合のみ停止します。条件はカスタム化された値を提供する、MapForce ビルドインライブラリ関数の書式を取ります。すなわち、条件が true を返す場合、ブレークポイントはマッピングの実行を停止します。

ブレークポイントを作成するには、以下を行います:

- 入力または出力コネクタ(コンポーネントの右または左横にある小さな三角形)を右クリックして、、**「デバッガー ブレークポイント」**を選択します
- 入力または出力コネクタをクリックして、 **「F9」**を押します。

条件付ブレークポイントの作成:

1. コネクタを右クリックして、 **「ブレークポイントプロパティ」**を選択します .



2. 両方の **「ブレークポイント」**を選択するために **「条件」**チェックボックスをクリックします。
3. リストから必要な関数を選択し、関数の値を、適用できる場合は、入力します。例:上のサンプルでは、ブレークポイントは、通過する値が2より大きい場合、マッピングの実行を停止します。

条件ブレークポイントを追加するコネクタのデータ型が、関数により期待される型と一致しない場合、MapForce は、データ型を自動的に変換しようと試みます。自動変換が無理な場合は、マッピングの実行は、失敗します。 これを回避するために、互換性のあるデータの型を使用するようにしてください。例 :関数 `core.starts-with` は、文字列の値を期待します、ですからブレークポイントのコネクタは同じ型を有する必要があります。

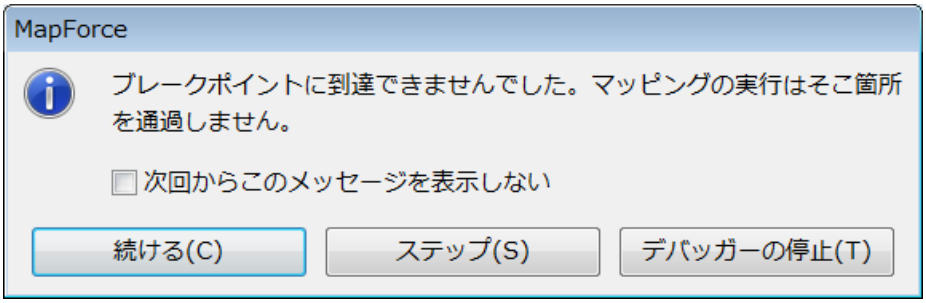
ブレークポイントの削除

ブレークポイントを削除するには、ブレークポイントが存在するコネクタを右クリックして、 **「デバッガーブレークポイント」**を選択します。 または、 ブレークポイントが存在する入力または出力コネクタをクリックし、 **「F9」**を押します。

ブレークポイントウィンドウからブレークポイントを削除することもできます（ブレークポイントウィンドウの使用 を参照 ）

到達することのできないブレークポイント

MapForce は、"ブレークポイントに到達することができません" メッセージを表示する場合があります ：



これは、以下の理由のためブレークポイントにデバッガーが到達できないことを意味します：

- マッピングの一部ではないコネクタでブレークポイントが定義されている場合。
- 実行の最適化のためMapForce がブレークポイントに到達できない場合 (制限を参照)。

次の定義されたブレークポイントに到達するために、**続ける** をクリックします。次に定義されたブレークポイントに移動する場合、デバッグの実行の終わりに移動する場合、デバッグをステップで実行する場合、**ステップ** をクリックします。

デバッガーが直面する到達することのできないブレークポイントに関する通知を無効化することが、**このメッセージを再度表示しない** をクリックすることにより設定できます。手順は以下のとおりです：

1. **「ツール」** メニューから **「オプション」** をクリックします。
2. **「メッセージ」** をクリックします。
3. **到達することのできないブレークポイントに関する情報** チェックボックスのチェックを解除するためにチェックボックスをクリックします。

6.4 値ウィンドウの使用

値ウィンドウは、デバッグモード中にマッピングにより処理される値についての情報を表示します。値ウィンドウに表示される情報は、現在のデバッガーのポジション、およびクリックするユーザーインターフェイスにより異なります。値ウィンドウには、以下のタブが含まれます：

"コンテキスト"タブ

「コンテキスト」タブは、現在処理されている値を表示します。（コンテキストウィンドウで同じ値のコンテキストが表示されています）。これは、デバッガーの現在の実行のポジションの値、または、過去に処理されたコネクタの値を表示します。MapForce はこれらの2つの違いを異なる色を使用して区別します：

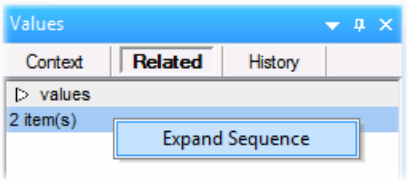
	緑は "現在 "を意味し、現在の実行のポジションを示します（コネクタの現在の値を表示するを参照してください）。
	黄色は "過去 "を意味し、現在の実行のポジションに相対して、過去のコネクタを観察していることを示します。これは、コンテキストを手動で設定した後に発生する場合があります（コンテキストを値に設定するを参照してください）。

"関連した" タブ

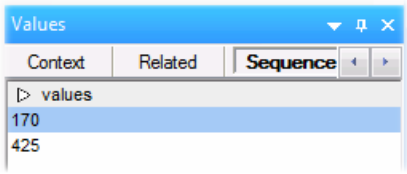
関連した タブは、関連した値、または、現在処理されている（または、"近い過去 "の値）を表示します。通常、このタブを明示的にクリックする必要があります。MapForce は自動的にデバッガーの現在の実行のポジションに関連したコネクタのデータオーバーレイをクリックすると切り替えます。最近の過去にステップバックを参照してください。

"シーケンス" タブ

存在する場合、**シーケンス** タブは、シーケンスを処理するコネクタの値にアクセスすることを有効化します。このタブはコネクタがアイテムのシーケンスを処理した場合のみ表示されます（例：`sum` または `count` などの関数）。アイテムのシーケンスを処理したコネクタのデータオーバーレイをクリックすると、値ウィンドウは、フォーマットで "**n アイテム**" エントリを、n がコネクタにより処理されたアイテム数である箇所で表示します。それぞれの値にアクセスするには、このエントリをダブルクリックします。（または右クリックして、コンテキストメニューから**シーケンスの展開**を選択します）。



値はシーケンスタブ内に表示されます。



履歴 タブ

履歴」タブは、デバッグが開始してから現在の実行のポジションまでに処理された値を表示します。 コネクタにより処理された値の履歴を確認するを参照してください。

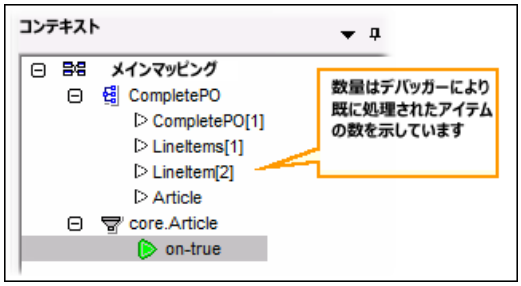
6.5 コンテキストウィンドウの使用

MapForce がデバッグモードの場合、コンテキストウィンドウは、デバッガーの現在のポジションに対して関連のあるコネクタの構造を表示します。すなわち、現在のマッピングの値を作成するマッピングコンテキストを与えます。

MapForce は、現在のコンテキストを以下の通り作成します：

1. ターゲット構造のルートノードから開始します。
2. 現在のターゲットノードに降順します。
3. 現在のターゲットノードから、マッピング内で現在のポジションにリードするコンポーネントを左側に移動します。これらのコンポーネントは、コンポーネント、ビルドインまたはユーザー関数、変数などをフィルターおよび並べ替える可能性があります。

コンテキストウィンドウは、情報およびナビゲートのを助ける役割を果たします。マッピング内の現在のコンテキストから直接 マッピング内で特定のノードを選択するには、コンテキストウィンドウ内のノードを右クリックして、**「マッピングの選択」**をクリックします。これはマッピングが大きいとスクロールを減らすことにより役に立ちます。
コンテキストウィンドウ以下の特別なアイコンと表記を表示する場合があります：

アイコン	説明
	コンテキストが属するマッピングを示します。これはメインマッピングまたはユーザー定義関数のマッピングであることができます。 
▷	コネクタを表します。ターゲットノードは角かっこで表示されています。 
	現在のコネクタ（最も最近の実行ポジション）を表しています。これは値ウィンドウ内の現在の値のソースです。 稀に、計算された値が複数のコネクタのために使用される場合があります。この場合、複数の緑色のアイコンが表示される可能性があります。
	デバッガーが最も最近の実行ポストに相対的な過去に存在する場合を表示します。これは値にコンテキストを設定すると生じる可能性があります（コンテキストを値に設定するを参照）。

上のアイコンに加え、コンテキストウィンドウは、マッピング内に存在するコンポーネントの標準アイコンを含みます。

コンテキストウィンドウとユーザー定義関数

現在のコンテキストがユーザー定義関数を含む場合、これらはコンテキストウィンドウにも表示されます。 Note that if 現在のコンテキストがユーザー定義関数の入力の値を計算するための場合、コンテキストは以下のように決定されます：

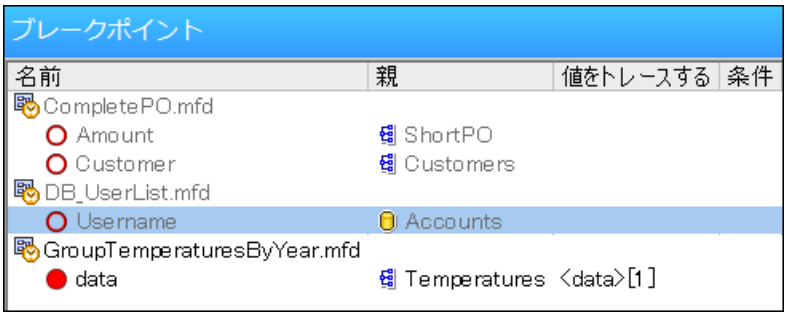
1. ターゲットからユーザー定義関数の出力コネクタ ユーザー定義関数の入力コネクタへ。
2. 更に左側に。

メモ： ユーザー定義関数は、コンテキスト内で複数回生じる可能性があります。これは、関数の呼び出しの一部がチェーンされている、またはユーザー定義関数が再帰的と定義されているためです。

6.5.1 ブレークポイントウィンドウの使用

ブレークポイントを使用することによりブレークポイントをグローバル的に表示し管理することができます。デフォルトでは、MapForce がデバッグモードの場合、ブレークポイントウィンドウは表示されます。ブレークポイントウィンドウを常に表示するには、**表示 | デバッグウィンドウ | ブレークポイント** を選択します。

ブレークポイントウィンドウは、MapForce を開始してから作成された全てのブレークポイントをマッピングファイルごとにグループ化し表示します。MapForce が開かれていると、マッピングに関連する全てのブレークポイントは MapForce により記憶されており、ブレークポイントウィンドウで表示されます。現在デバッグされているマッピングは標準テキストの色でブレークポイントウィンドウで表示され、他のマッピング (閉じられているまたは非アクティブなマッピング) は灰色で表示されています。



ブレークポイントウィンドウ内の全てのマッピング (またはその中のブレークポイント) をダブルクリックすることで素早く開くことができます。

メモ： MapForce を閉じる、または、再起動すると、すべてのブレークポイントは削除されます。

ブレークポイントに関する情報は下の表にまとめられています：

カラム	説明
名前	ブレークポイントが属するノードの名前
親	ブレークポイントが属するマッピングコンポーネントの名前
トレースする値	ブレークポイントで接続をパススルーする値。トレースする値は、値デバッグの実行中に表示されます。
条件	ブレークポイントに条件がある場合、この列はブレークポイントの条件を表示します。

ブレークポイント次のアイコンに関連している可能性があります。

アイコン	説明
	アクティブなブレークポイント。現在デバッグされているブレークポイントをマッピングから表します。
	非アクティブなブレークポイント。現在デバッグされていない、開かれているブレークポイントをマッピングから表します。
	到達することのできないブレークポイント。デバッガーにより到達することのできないブレークポイントを表します。
	条件付のブレークポイント。条件付のブレークポイントを表します。

ブレークポイントのプロパティを表示または変更する方法：

- 右クリックして、コンテキストメニューから、**「ブレークポイントプロパティ」**を選択します

ブレークポイントを削除する方法。：

- 削除するブレークポイントを右クリックして、コンテキストメニューから**「ブレークポイントの削除」**を選択します。
- ブレークポイントをクリックして、**「削除」**を押します。

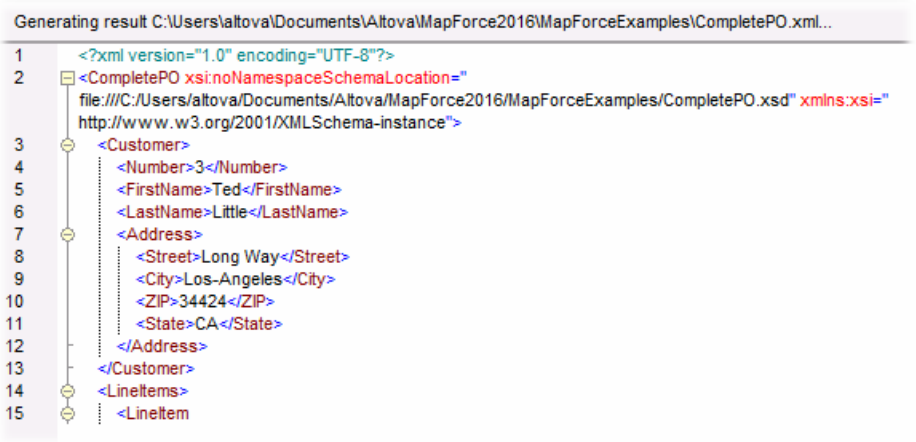
コンテキストコマンド**「全てのブレークポイントを削除」**は、マッピング内の存在箇所に関わらずブレークポイントウィンドウ内に表示される全てのブレークポイントを削除します。

以下を参照してください：ブレークポイントの追加と削除

6.6 部分的に生成された出力のプレビュー

段階的に、またはブレークポイントを使用してデバッグを行う場合、現在のデバッガーのポジションまでの出力されたマッピングの出力を確認することができます。生成された出力の一部のプレビューはXML、flat text、およびEDI ターゲットコンポーネントによりサポートされています。

デフォルトでは、 **F5**」を押すと（ブレークポイントを定義せずに）MapForce は、デバッグモードで全体のマッピングを実行し、 **出力**」 タブに切り替え、最終的に生成された出力を表示します。しかしながら、ブレークポイントを定義している場合、または、デバッグを段階的に行っている場合（**F11** または **Ctrl + F11**）デバッガー実行はマッピング出力が生成されているにもかかわらず停止されます。マッピング出力が部分的に書かれている場合でも、 **出力**」 タブをクリックすることができ、また、プレビューすることができます。



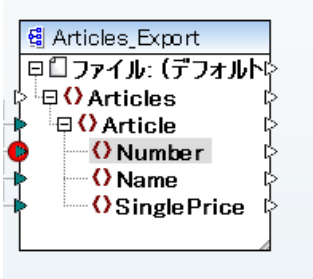
制限

- 現在計算されるターゲットノードは、 出力」 タブに常に表示されているわけではありません。例 :XML 属性は内部で集められ、一度に書き込まれます。
- 出力が複数のファイルを作成する場合、現在書き込まれているファイルのみが表示されます。他の出力ファイルに切り替えると無効化されます。

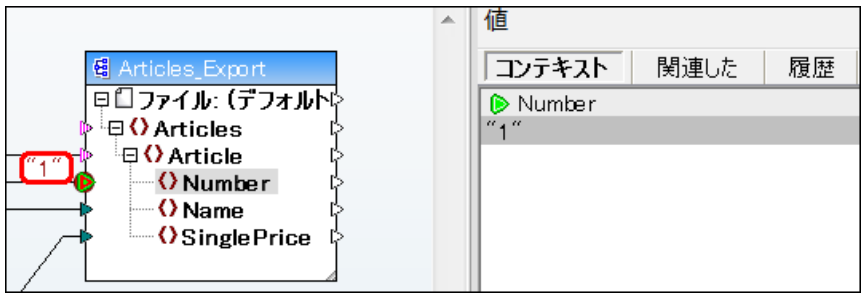
6.7 コネクタの現在の値を表示する

(ステップでデバッグを実行しているか、コネクタ上にブレークポイントが定義されているため)デバッガーの現在の実行のポジション（ ）が特定のコネクタ上にある場合、コネクタにより処理される現在の値は値ウィンドウの **「コンテキスト」** タブに表示されます。この値が出力に書き込まれようとしている値であり、すなわち、現在 '的值になります。これはまたコンテキストウィンドウにコンテキストが表示される値でもあります（コンテキストウィンドウの使用を参照）。

このシチュエーションを理解するには、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリから**PreserveFormatting.mfd** を開きます。ターゲットコンポーネントの`Number` ノードの入力コネクタをクリックし、**F9** を押して、ブレークポイントを追加します。



次に、**F5** を押して、デバッグを開始して、結果全体を確認します。



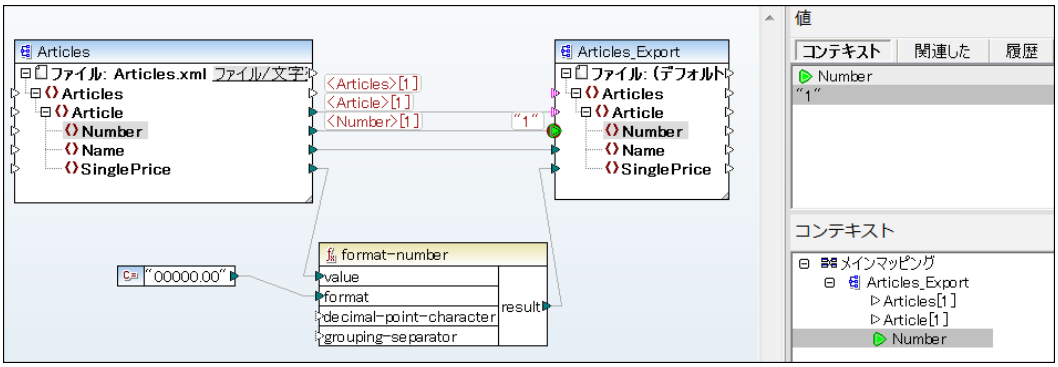
イメージで表示されるように、現在のデバッガーのポジション（およびブレークポイント）ターゲットコンポーネントの`Number` ノード上にあります。値ウィンドウはこのノードが値 "1"を有することを示しています（この値もマッピング上では、赤い縦じまハイライトされています）。

6.8 最近の過去にステップバックする

マッピングコネクタの横のデータオーバーレイ(小さな表方形のボックス)をクリックすると、**値** ウィンドウは名前、とオプションで、選択されたコネクタに関連した値を表示します。 フォーカスは現在のデバッガーのポジションでは無く、選択されたデータオーバーレイです。このビューをデバッグの履歴を少し戻ったビューと考えることができます。マッピングは、現在のデバッガーのポジションに関連した最後のいくつかのコネクタのためのオーバーレイのみを表示するため、これは、 "近い"過去です。関連した"データオーバーレイをクリックすると、値 ウィンドウは自動的に**関連した** タブに切り替えます。

このシナリオの説明に関しては、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリからマッピング**PreserveFormatting.mfd** を開いてください。

マッピングを開いた後、ターゲットコンポーネントのNumber ノードの横のコネクタをクリックし、 **「F9」**を押して、ブレークポイントを追加します。 **「F5」**を押してデバッグを開始します。ソースコンポーネントのNumber ノードの横にあるデータオーバーレイ (小さな表方形のボックス)をクリックします。



マッピング内でコネクタは、通常、複数回反復されるため、反復の現在のインデックスは各かっこ内に表示されます : **<Number> [1]**。また、コネクタは値を持ち、その値がまた等号サインの後に以下のように表示されます :**<Number> [1]=1**。同じ値は以下のように値ウィンドウの新規の行に表示されます。

特定の値についての追加情報が必要な場合、その値を生成したコンテキストを再作成できることに注意してください (コンテキストを値に設定する を参照)。

6.9 コネクタにより処理された値の履歴を確認する

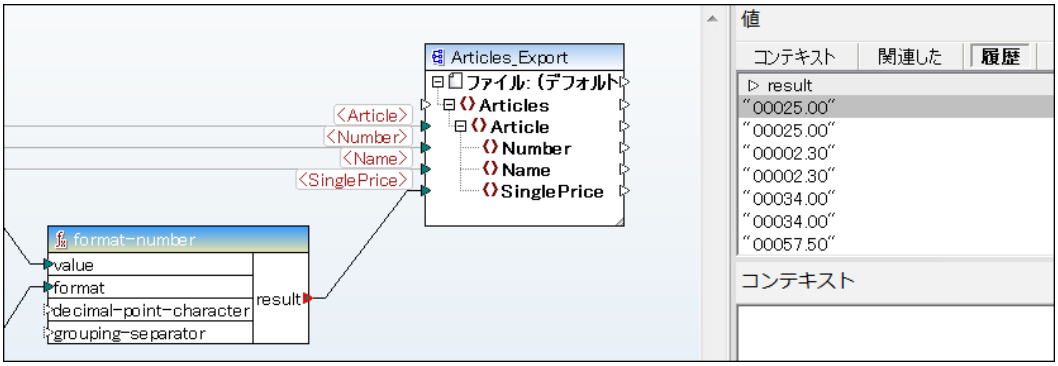
オプション **「トレースの履歴の全体を保管」** が有効化されている場合、(現在の実行のポジションまで)そのコネクタにより処理された全ての値の履歴をビューすることができます (デバッガー設定 を参照)。

コネクタをクリックすると、履歴が表示されます。値ウィンドウの **「履歴」** タブをクリックします。このオペレーションはマッピングの実行の開始から、現在のデバッガーのポジションまでの間、値を処理したコネクタに対してのみ意味があることに注意してください。

このケースを説明するには、ブレークポイントを使用せず、始めからマッピングをデバッグし、特定のコネクタにより処理されたすべての値の履歴を確認してみましょう。最初に、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリからマッピング **PreserveFormatting.mfd** を開きます。既に開かれている場合は、以下を行います:

- ブレークポイントが存在する場合、全てクリアします (ブレークポイントの追加と削除を参照)。
- **Shift + F5** を押すと、現在実行中のデバッグを中止することができます。

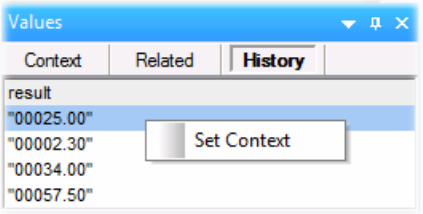
準備ができると、**「F5」** を押して、新規のデバッグの操作を開始します。**「F5」** を押すと、MapForce はマッピングをデバッグモードで実行し、**「出力」** タブに切り替えます。**「マッピング」** タブをクリックして、メインマッピングウィンドウ `format-number` 関数の `result` ノードに移動します (下のイメージで赤でハイライトされています)。最後に、値ウィンドウの **「履歴」** タブをクリックします。表示される値を確認してください。



上のイメージで表示されるとおり、この特定のノード (`result`) は全部で4つの値を処理しました。特定の値についての追加の情報が必要な場合、その値を作成したコンテキストを再作成することができることに注意してください (コンテキストを値に設定する を参照)。

6.10 コンテキストを値に設定する

コンテキストの値に設定することは、その値を生成したマッピングコンテキストを詳細に確認する、過去にさかのぼり比較するアクションです。コンテキストを**関連した**タブ、**シーケンス**タブ、または**「履歴」**タブ内の値ウィンドウに表示される値に設定することができます。**「トレースの履歴の全体を保管」**オプションを有効化した場合 (デバッガー設定 を参照)、**「履歴」**タブは、現在選択されているコネクタにより処理されたすべての値を表示します。 ですから、この場合、追加して、そのコネクタのために過去にさかのぼりコンテキストの値を設定することができます。

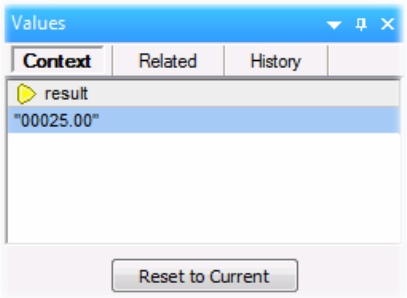


コンテキストを値に設定するには、以下を行います：

- 値を右クリックして、コンテキストメニューから、**「コンテキストの設定」**を選択します
- 値をダブルクリックします。

値をコンテキストに設定すると値を生成した状況を再作成し、**値**ウィンドウと**コンテキスト**ウィンドウを選択したコンテキストに従い作成するために、MapForce は、マッピングエリアをハイライトします。コンテキスト内のマッピングで使用されたビジュアルヒントのレジェンドに関しては、デバッグモードについてを参照してください。コンテキスト自身についての詳細は、コンテキストウィンドウの使用を参照してください。

手動で設定されたコネクタは、最も最近の実行ポジションでないことを示す黄色 (▷) です。最も最近の実行ポジションに切り替えるには、(適用できる場合)、値ウィンドウの**「コンテキスト」**タブから、**「現在にリセット」**ボタンをクリックします。



6.11 デバッガー設定

MapForce デバッガーに適応することのできる設定へのアクセスの方法は、以下のとおりです。メニューコマンド **「ツール | オプション」** を選択して、から **「デバッガー」** をクリックします。 使用することのできる設定は、以下のとおりです：

値の最高ストレージの長さ：

値ウィンドウ内で表示される文字列の長さの値を定義します（最低 15 文字）。ストレージの長さを高い値に設定すると、使用可能なシステムメモリを消耗する点に注意してください。

トレースの履歴の全体を保管

MapForce に、デバッグ中マッピング内の全てのコンポーネントのコネクタにより処理された全ての値を保管するように命令します。このオプションが有効化されていると、デバッグの始めからMapForce により処理された全ての値が、メモリ内に保管され、デバッグを中止するまで、ウィンドウ内で分析のために使用することができます。データ集中型マッピングをデバッグする場合にこのオプションを有効化することは、デバッグの実行を遅らせ、使用できるシステムメモリを消耗するため奨励されません。このオプションが無効化されている場合、MapForce は、現在の実行のポジションに相対するノードの最も新しいトレース履歴のみを保管します。

Chapter 7

データソースとターゲット

7 データソースとターゲット

このセクションは、MapForce がマッピングに必要なソースとターゲットコンポーネントの型について説明します。

- XML とXML スキーマ
- データベースとMapForce
- CSV のマッピングとテキストファイル
- MapForce Flextext
- EDI - UN/EDIFACT、ANSI X12、HL7、IATA PADIS、SAP IDoc
- JSON
- Microsoft OOXML Excel 2007+
- XBRL
- HL7 v3.x からXML スキーマへ

(C) 2017 Altova GmbH

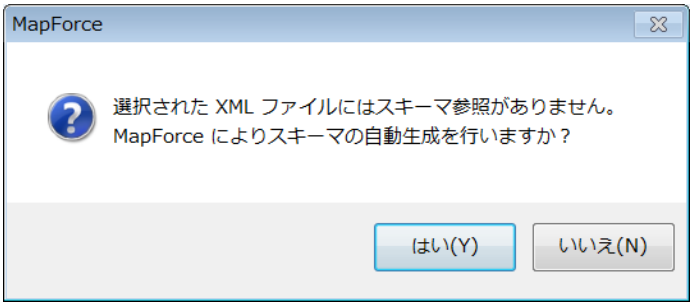
7.1 XML と XML スキーマ

このドキュメントのはじめに、XML とXML スキーマファイルをソースまたはターゲットコンポーネントとして、使用する簡単なマッピングのサンプルについて説明しました。このセクションはマッピング内でXML コンポーネントの使用についての詳細について説明します。以下のトピックが含まれます：

- XML コンポーネント設定
- DTD をスキーマコンポーネントとして使用
- 派生したXML スキーマ型とマッピング
- QName サポート
- Nil 値 /Nillable
- コメントと処理命令
- CData セクション
- ワイルドカード -xs:any

7.1.1 XML スキーマの生成

スキーマがない場合、MapForce は、自動的にXML スキーマを既存のファイルをベースに生成することができます。マッピングエリアにスキーマを持たないXML ファイルを追加すると、次のダイアログボックスが表示されます。

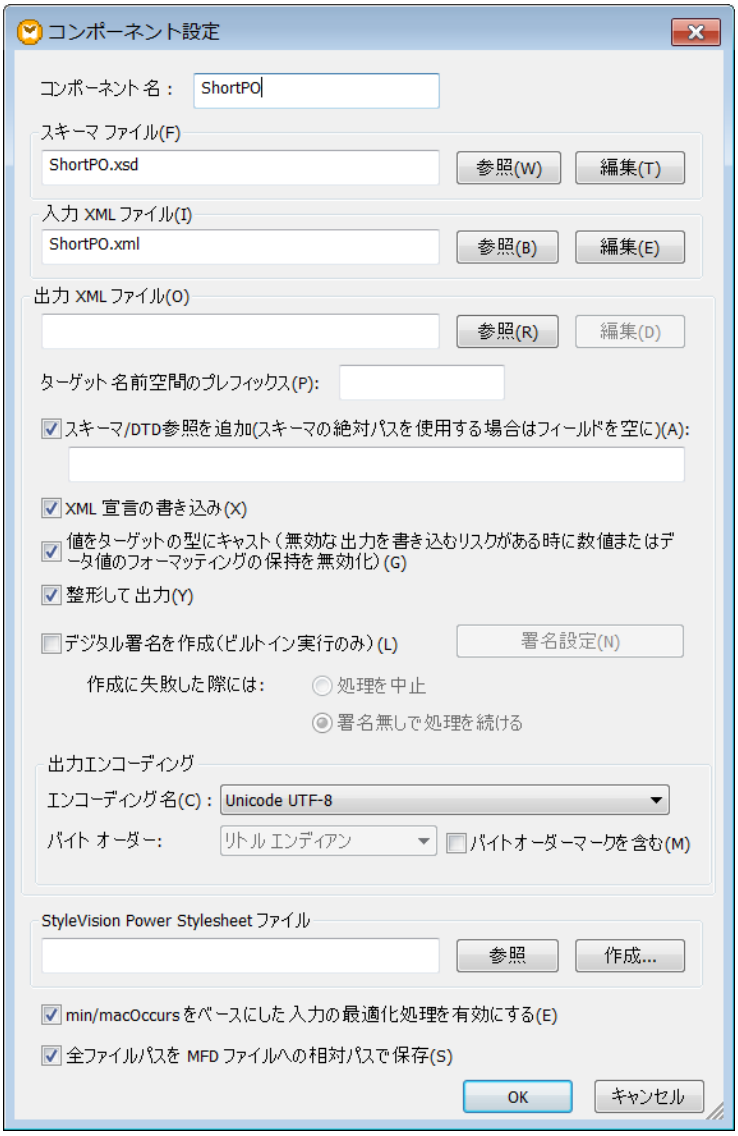


MapForce がXML ファイルからスキーマを生成する場合、要素 属性は、XML インスタンスドキュメントから推定されなければならず、予期しないものの可能性があります。生成されたスキーマがインスタンスデータを正確に表していることを確認することが奨励されます。

7.1.2 XML コンポーネント設定

マッピングエリアにXML コンポーネントを追加した後、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます：

- マッピング上のコンポーネントを選択し、 **「コンポーネント」** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。



XML コンポーネント設定 ダイアログボックス

使用することのできる設定は、以下のとおりです .

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名が自動的に生成され、*インスタンスファイル*をその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。 コンポーネント名 はスペース(例 : "ソースXML ファイル") またフルストップ文字も含むことができます (例 : "Orders.EDI")。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭および末尾スペースを含んではなりません。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください :

	• マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
スキーマファイル	MapForce によりデータを検証とマップするために使用された XML スキーマファイルの名前とパスを指定します。 スキーマファイルを変更するには、 **参照]** をクリックして、新しいファイルを選択します。XMLSpy 内のファイルを編集するには、 **編集]** をクリックします。
入力 XML ファイル	MapForce がデータを読み込む XML インスタンスファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初にコンポーネントを作成した際に入力され、XML インスタンスファイルに割り当てられます。 ソースコンポーネント内で、インスタンスファイル名は XML ルート要素とレファレンススキーマを検出するため、また選択されたスキーマに対して検証するために使用されます。 ファイルのロケーションを変更するには、 **参照]** 新しいファイルを選択します。XMLSpy 内のファイルを編集するには、 **編集]** をクリックします。
出力 XML ファイル	MapForce がデータを書き込む XML インスタンスファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 ファイルのロケーションを変更するには、 **参照]** をクリックして、新しいファイルを選択します。XMLSpy 内のファイルを編集するには、 **編集]** をクリックします。
ターゲット名前空間のためのプレフィックス	ターゲット名前空間のためにプレフィックスを入力することを許可します。プレフィックスを割り当てる前に、ターゲットスキーマ内でターゲット名前空間が定義されている必要があります。
スキーマ/DTD レファレンスの追加	参照された XML スキーマファイルを XML 出力のルート要素に対してパスを追加します。このフィールドに入力されたスキーマのパスは、`xsi:schemaLocation` 属性または DTD が使用される場合、`DOCTYPE` 宣言 内の生成されたターゲットインスタンスファイルに書き込まれます。 XQuery または C++ 出コードを生成する場合、DTD レファレンスの追加はサポートされません。 このフィールドにパスを入力することにより、XML インスタンスファイルにより参照されたスキーマファイルがどの箇所に位置するかを決定することができます。これにより、マッピングが実行される際、出力インスタンスがマッピングのデスティネーションで検証されることを保証できます。また、**http://** アドレスや絶対及び相対パスをこのフィールドに入力することもできます。 このオプションを無効化することにより、XML インスタンスを参照された XML スキーマまたは DTD から、切り離すことができます (例 :結果する XML 出力を基となる XML スキーマを持たない相手に送信する場合など)。
XML 宣言の書き込み	このオプションにより生成された出力から XML 宣言 を表示しないようにすることができます。デフォルトでは、オプションが有効化されていると、XML 宣言 が出力に表示されます。

<table>
<tr><td></td><td>この機能は MapForce ターゲット言語と実行エンジンでサポートされています。

<table>
<tr><th>ターゲット言語 /実行エンジン</th><th>出力がファイルの場合</th><th>出力が文字列の場合</th></tr>
<tr><td>ビルドイン</td><td>はい</td><td>はい</td></tr>
<tr><td>MapForce Server</td><td>はい</td><td>はい</td></tr>
<tr><td>XSLT, XQuery</td><td>はい</td><td>いいえ</td></tr>
<tr><td>コードジェネレータ (C++、C#、Java)</td><td>はい</td><td>はい</td></tr>
</table></td></tr>
<tr><td>ターゲット型に値をキャストする</td><td>Aターゲット XML スキーマ型がマッピングで使用されるかを定義します。または ターゲットコンポーネントにマップされる全てのデータが文字列値として扱われるかを定義します。デフォルトでは、この設定 が有効化されています。

このオプションを無効化することにより、正確なフォーマットの値を保持することができます。例 : これは、数値内で特定の小数点を必要とするスキーマ内のパターンファセットを満足させるために この場合役にたちます。

マッピング関数を使用して、数値を文字列として必要とされるフォーマット内で使用し、この文字列をターゲットにマップすることができます。

このオプションを無効化にすると、無効な値の検出も無効化されます。例 数値フィールドに文字が入力された場合。</td></tr>
<tr><td>整形出力</td><td>出力 XML ドキュメントに構造化された外見を与えるために整形出力します。それぞれの子 ノードは、単一タブ文字のそれぞれの親のオフセットです。</td></tr>
<tr><td>デジタル署名の作成</td><td>XML 出力インスタンスファイルにデジタル署名を追加することを許可します。"ビルドイン"を変換言語として選択してもデジタル署名を追加することができます。 (デジタル署名も参照)。</td></tr>
<tr><td>出力エンコード</td><td>出力インスタンスファイルの次の設定を指定することを許可します :

• エンコード名
• バイトオーダー
• バイトオーダーマーク (BOM) 文字が含まれるか否か。

デフォルトでは、 新しいコンポーネントのためのデフォルトのエンコード オプション内で定義されたエンコードを持つ新しいコンポーネント。 [ツール | オプション] からこのオプションにアクセスし、タブを生成します。

マッピングが XSLT 1.0/2.0 を生成すると、バイトオーダーマーク チェックボックスを有効化しても、これらの言語はバイトオーダーマークをサポートしないため効果はありません。</td></tr>
<tr><td>StyleVision Power スタイルシートファイル</td><td>このオプションにより、Altova StyleVision スタイルシートファイルを選択または作成することができます。このようなファイルは、XML インスタンスファイルからのデータを HTML、RTF などの多種のレポートに適したフォーマットで出力することができます。</td></tr>
</table>

	以下も参照 : コンポーネント上で相対パスを使用する
min/maxOccurs をベースに入力処理の最適化の有効化	このオプションにより、`minOccurs` と`maxOccurs="1"` を持つ必要な属性または子要素 などの1つのアイテムのみを含むシーケンスを特別に処理することが許可されます。この場合、シーケンスの最初のアイテムが抽出され、アイテムは (シーケンスとしてではなく) 直接動的な値として処理されます 入力データがスキーマに対して場合、**有効でない場合、**エラーメッセージと共にマッピングを停止する空のシーケンスがマッピング内で生じる可能性があります。このような**無効な入力**を処理するには、このチェックボックスを無効化にします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。以下も参照 : コンポーネント上で相対パスを使用する。

7.1.3 "スキーマ" コンポーネントとしてを DTD 使用する

MapForce 2006 SP2 以降では、ソースならびにターゲットコンポーネントにて名前空間を意識したDTD がサポートされます。名前空間 URI がDTD の "xsIns" 属性宣言から抽出され、マッピングを行うことが可能になります。

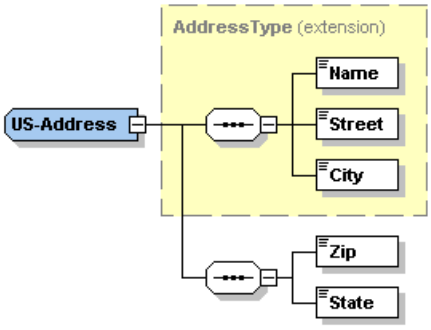
しかし StyleVision により使用される DTD の様に、名前空間 URI を持たずに xmlns* 属性を含む DTD も存在します。このような DTD は、MapForce で使用することができるように拡張する必要があります。以下に示されるように、DTD を修正して名前空間 URI を含む xmlns 属性を定義する必要があります:

```
<!ATTLIST fo:root
   xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
   ...
>
```

7.1.4 派生した XML スキーマの型

MapForce では複合型の派生型に対するマッピングがサポートされます。派生型は、**xsi:type** 属性により特定の派生型を指定するXML スキーマの複合型です。

以下のスクリーンショットは、XMLSpy にて派生型の "US-Address" の定義を示したものです。基底型 (またはオリジナルとなっている複合型) はこの場合 AddressType となります。`Zip` と`State` の2 つの要素を新たに追加することで、派生型のUS-Address が作成されます。



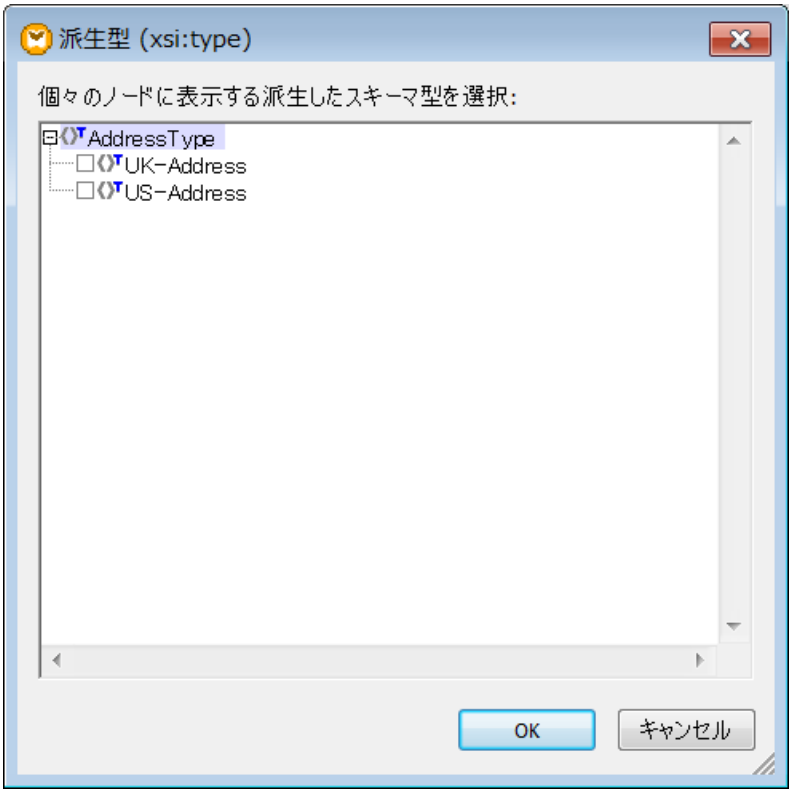
生成された型のサンプル (XMLSpy スキーマビュー)

次のサンプルは、派生したXML スキーマ型からデータをマップ、または、スキーマ型へのデータのマップの方法を説明しています。

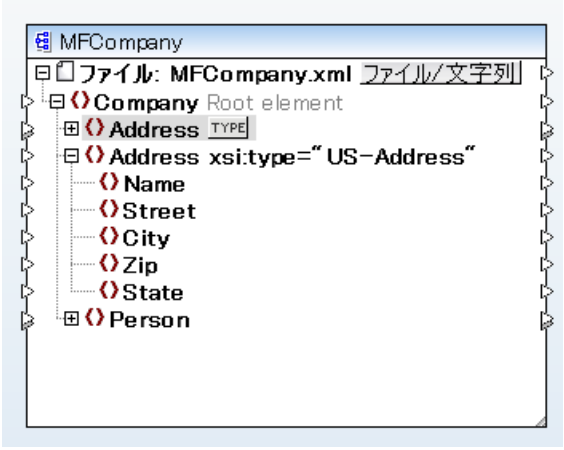
1. **「挿入」**メニューから**「XML スキーマ/ファイル」**をクリックします。以下のXML スキーマ:**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\MFCompany.xsd** を開きます
2. インスタンスファイルを提供するようにプロンプトされると、**「スキップ」**をクリックして、ルート要素をとしてCompany選択します。



3. Address 要素の横の TYPE ボタンをクリックします。このボタンは、スキーマ内のこの要素のために存在する派生した型であることを示しています。



4. 使用する派生した型の横のチェックボックスを選択し、 (この場合、US-Address)、 **「OK」**を押して確認します。 新規の要素 `Address xsi:type="US-Address"` がコンポーネントに追加されます。



これらのアイテムに対して直接、または、 US-Address から、マッピングを行うことが可能になります。

派生型ダイアログボックスにて複数のアイテムを選択することで、複数の派生型を追加/挿入することができます。各ノードにはそれぞれ独自のxsi:type 要素が与えられます。

7.1.5 QNames

MapForce は、マッピングの実行ランタイムでXML ファイルからデータを読み込む際にQName (修飾名) プレフィックス (http://www.w3.org/TR/xml-names/#ns-qualnames) を解決します。

QNames は、XML と XBRL インスタンスドキュメント内の名前空間 URI を参照および省略するために使用されます。QNames には、2つの種類があります :プレフィックスを持つ、またはプレフィックスを持たない QNames.

PrefixedName	Prefix ':'	LocalPart
UnPrefixedName		LocalPart LocalPart が要素 または属性の名前の箇所

例 :下のリストでは、`<x:p/>` がQName です :

- プレフィックス 'x' は名前空間 "**http://myCompany.com**" の省略形です。
- p は、ローカルパートです。

```
<?xml version='1.0'?>
<doc xmlns:x="http://myCompany.com">
   <x:p/>
</doc>
```

MapForce は、**xpath2 | qname-related 関数** ライブラリにQName に関連した関数の一部を含みます。

7.1.6 Nil の値 / Nillable

XML スキーマ仕様に従えば、nillable="true" 属性をスキーマ内の要素に対して指定することで、(コンテンツのデータ型により空のコンテンツが許されていない場合でも) コンテンツを含まない要素でも妥当とみなすことができるようになります。XML インスタンスドキュメント内に記述される属性の xsi:nil="true" により、要素は存在するが、コンテンツは含まれない (つまり空要素である) ことが示されます。このセクションはどのように、MapForce がソースとターゲットコンポーネント内の nil 要素を扱うかを説明します。

'xsi:nil' と 'nillable' の比較

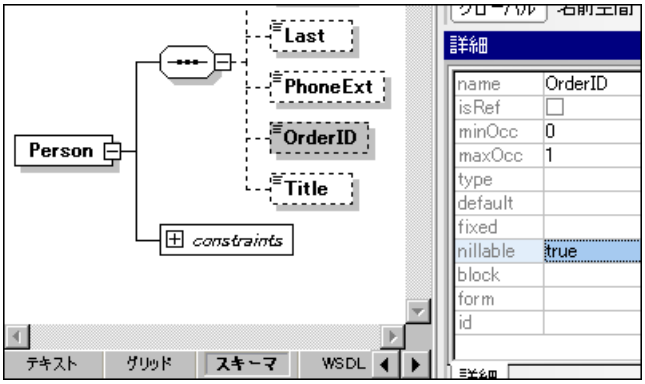
xsi:nil="true" 属性はXML **インスタンス**ファイルにおいてのみ定義されます。

```
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>
```

xsi:nil 属性はMapForce のマッピングにて視覚的 (かつ明示的) に示されることは無く、自動的に処理されます。通常、null 値が許容されたノード (xsi:nil="true" 属性が含まれるノード) は存在するが、コンテンツには何も含まれない状態になります。

`xsi:nil` 属性は、通常自動的に処理されるため、MapForce グラフィカルなマッピングで明示的に表示されません。特に "nilled" ノード (`xsi:nil="true"` 属性を持つ) が存在する場合、そのコンテンツは存在しません。

`nillable="true"` 属性は、XML **スキーマ**内で定義されます。MapForce では、ソースとターゲットコンポーネント内に存在することができます。



マッピングソースにある null 値が許容された要素

マッピングにて null 値を許容する (nillable な) データを XML または XBRL 要素から読み取った場合、xsi:nil 属性が自動的にチェックされます。xsi:nil の値が true となっている場合、コンテンツは存在しないものとして扱われます。

nillable なソース要素から**単純コンテンツ**の (属性は含まれるが子要素を含まない) nillable なターゲット要素に対して、**ターゲット優先**マッピングでマッピングを作成し、xsi:nil がソース要素にセットされている場合、xsi:nil 属性がターゲット要素にも挿入されます (例 :`<OrderID xsi:nil="true"/>`)。

全てコピーマッピングを nillable なソース要素から nillable なターゲット要素へマッピングして、xsi:nil がソース要素にてセットされている場合、xsi:nil 属性がターゲット要素にも挿入されます (例 :`<OrderID xsi:nil="true"/>`)。

ソース要素の xsi:nil に true がセットされているかを明示的にチェックするには、`is-xsi-nil` 関数を使用します。「nil 化」された要素に対して true が返され、それ以外のノードに対して false が返されます。

nil 化された (存在しない) ソース要素の値を別の値で代用するには、`substitute-missing` 関数を使用してください。

メモ:

- コンテンツが無くても要素ノードが実際存在するため、`exists` 関数を nill 化されたソース要素 に接続すると TRUE を返します。
- `xsi:nil` が設定されている箇所で (`multiply` と`concat` などの) 単純型の値を期待する関数を要素に対して使用すると、要素コンテンツが存在しなく、結果を取得することはできません。これらの関数はソースノードが存在しないかのように振舞います。

マッピングターゲットとしての Nillable 要素

nillable なソース要素から**単純コンテンツ**の (属性は含まれるが子要素を含まない) nillable **なターゲット要素に対して、ターゲット優先**マッピングでマッピングを作成し、xsi:nil がソース要素にセットされている場合、xsi:nil 属性がターゲット要素にも挿入されます (例 :`<OrderID xsi:nil="true"/>`)。 xsi:nil="true" 属性が XML ソース要素にセット**されていない**場合、通常の方法で要素コンテンツがターゲット要素へマッピングされます。

マッピングが (子要素を持つ) **複合型**の nillable 要素に対して行われている場合、xsi:nil 属性が自動的に書き込まれることはありません。これはマッピングが行われた後で子要素が書き込まれるか分からない為です。**全てコピー**接続を定義することで、ソース要素から xsi:nil 属性をコピーすることができます。

空のシーケンスまたはデータベースの NULL 値をターゲット要素へマッピングする場合、それが nillable であるかどうかにかかわらず、要素が作成されることはありません。

空のターゲット要素を xsi:nil="true" 属性がセットされた状態で (強制的に) 作成するには、`set-xsi-nil` 関数を使

ーゲット要素へ直接接続してください。この方法は単純型ならびに複合型のターゲット要素に対して使用することができます。

substitute-missing-with-xsi-nil 関数を使用することで、マッピングソースから値を得ることができない場合に xsi:nil をターゲットへ挿入することができます。これはソースノードが全く存在しない場合、または (multiply のような) 計算に対して nil 化されたソースノードが与えられ、結果が得られなかった場合などに使用することができます。

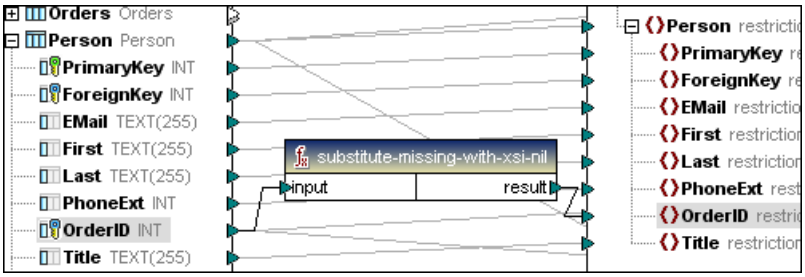
この関数はデータベースの NULL 値から xsi:nil="true" を持つ要素を作成するのに使用することができます。この関数は単純型のコンテンツを持つノードに対してのみ使用することができます。

> **メモ:**
> - `xsi:nil` を生成する関数は、(**if-else** 関数などの) 値に対して動作する関数またはコンポーネントにパススルーされることはできません。

NULL データベースフィールドから xsi:nil へマッピング

NULL データベースフィールドが XML スキーマの nillable 要素へマッピングされた場合、実際にデータを含んでいるフィールドに対応したターゲット要素だけが生成されます。NULL データベースフィールドの要素はターゲットコンポーネントにて作成されません。

ターゲットコンポーネント内ですべての要素の作成を強制するには、*アライブラリ*のノード関数から**substitute-missing-with-xsi-nil** 関数を使用します。



上のスクリーンショットは、どのように**substitute-missing-with-xsi-nil** 関数が全てのデータベースフィールドのためにターゲット要素を作成するかを示しています:

- ターゲット要素内で<OrderID xsi:nil="true" /> を含むすべての不足している/NULL データベースフィールド
- データベースフィールドがターゲット要素に直接マップされる既存のデータ例 :<OrderID>1</ OrderID>.

データベースコンポーネントの Null フィールドを確認するには、**データベースクエリ**ボタンをクリックして、データベーステーブルに対してクエリを実行します。Null フィールドは「結果」ウインドウ内で「NULL」と表示されます。

	EMail	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

xsi:nil を NULL データベースフィールドにマップする

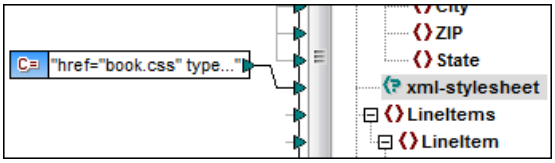
nill 化されたXML 要素をデータベース列にマップするには、MapForce はNULL 値をデータベースに書き込みます。
`set-null` 関数を使用して、データベースフィールドをNULL に無条件で設定することもできます。

7.1.7 コメントと処理命令

コメントと処理命令はターゲットXML コンポーネントに挿入することができます。処理命令は、XML ドキュメントを更に処理するアプリケーションに情報をパスするために使用されます。コメントと処理命令は、全てコピーマップグループのためのノードのためには定義することができないことに注意してください。

処理命令の挿入方法 :

1. ターゲットコンポーネント内の要素を右クリックして、コメント処理命令を選択し、メニュー (前、後) から処理命令オプションを選択します。
2. 処理命令 (ターゲット) 名をダイアログ内に入力し、「OK」を押して確認します。例 :xml-stylesheet。これによりこの名前をコンポーネントツリーに追加します。

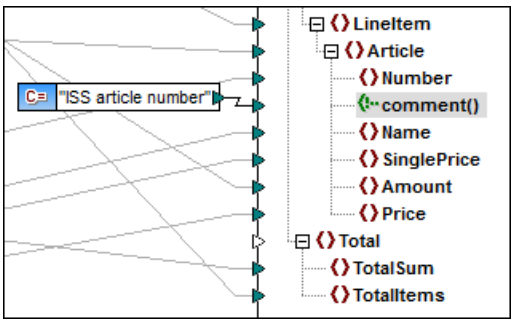


3. 以下を使用することができます。例 :処理命令属性の値を与えるために定数コンポーネントを使用する。例 :
`href="book.css" type="text/css".`

メモ:
複数の処理命令は、ターゲットコンポーネント内の要素の前またはお後ろに追加することができます。

コメントの挿入 :

1. ターゲットコンポーネント内の要素を右クリックして、コメント処理命令を選択し、メニューからコメントオプション (前、後ろ) を選択します。



これは、コンポーネントツリーにコメントノード (`<!--comment()`) を追加します。
2. 定数コンポーネントを使用して、コメントコンテキストを与えます、または ソースノードをコメントノードに接続します。

メモ:
単一のターゲットノードには1つのコメントのみを追加することができます。複数のコメントを作成するには、入力関数の複製を使用します。

コメント/処理命令の削除：

- 対応するノードを右クリックして、コメント/処理命令を選択し、フライアウトメニューから「削除」コメント/処理命令を選択します。

7.1.8 CDATA セクション

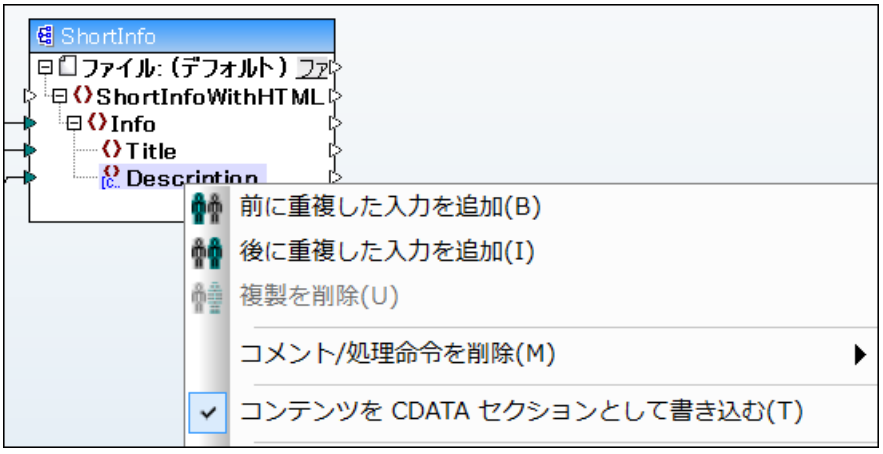
CDATA セクションは、通常マークアップとして解釈される文字を含むテキストのブロックをエスケープするために使用されます。CDATA セクションは "<![CDATA[" で開始し、"]]>" で終わります。

ターゲットノードは、CDATA セクションとして受信される入力データを書き込むことができます。ターゲットノードコンポーネントは以下であることができます：

- XML データ
- データベースフィールド内に埋め込まれた XML データ
- XBRL ターゲット内の型指定されたディメンションの XML 子要素

CDATA セクションの作成：

1. CDATA セクションとして定義するターゲットノードを右クリックして、"CDATA セクションとしてコンテンツを書き込む"を選択します。



入力データは、区切り記号 "]]>" で終了された CDATA セクションを含まない用に警告プロンプトを表示します。「OK」をクリックしてプロンプトを閉じます。
下に表示される [C.. アイコンでは、要素タグがこのノードが CDATA セクションとして定義されていることを示しています。



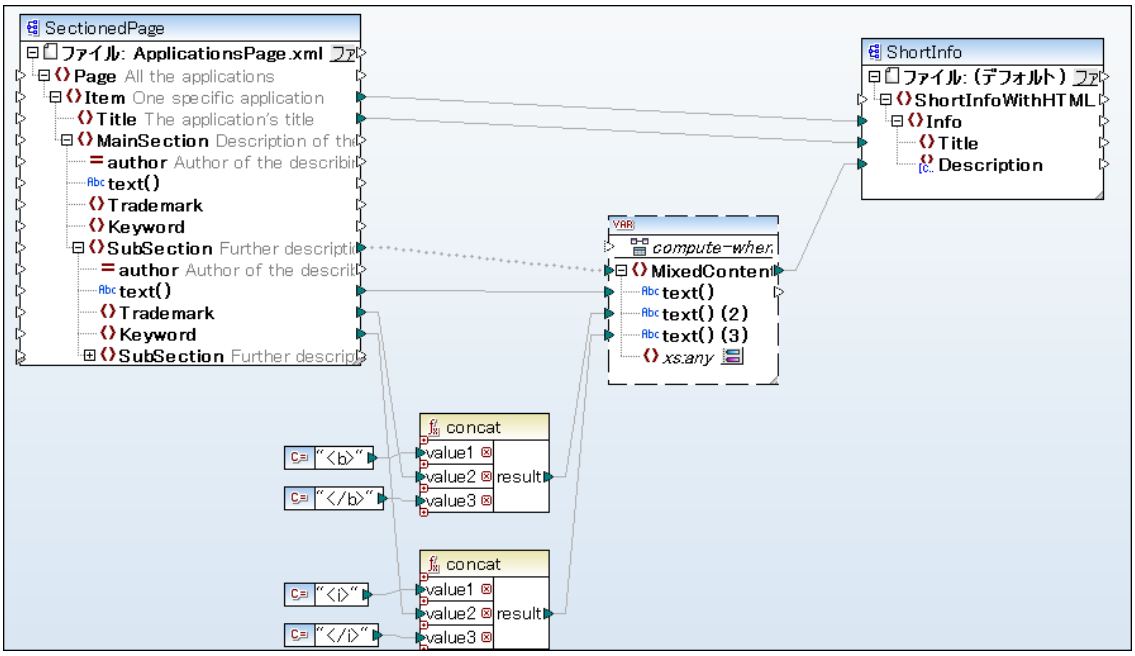
メモ:
CDATA セクションは、複製されたノード、および、xsi:type ノード上でも定義されることができます。

例：
...\MapForceExamples フォルダー内で使用することのできる **HTMLinCDATA.mfd** マッピングファイルは、CDATA セクションが有効利用できる例を表示しています。

この例：

- 太字の開始 () と終了 () タグは **Trademark** ソース要素のコンテンツに追加することができます。

- イタリックの開始 () と終了 () タグは**Keyword** ソース要素のコンテンツに追加することができます。
- 結果のデータは、サブセクション要素コネクタがソース優先 (複合型コンテンツ) ノードとして定義されているため、ソースドキュメント内に表示されるために複製 **text()** ノードにパスされます。
- 複合型ノードの出力は、CDATA セクションとして定義されているShortInfo ターゲット コンポーネント内の**詳細**ノードにパスされます。

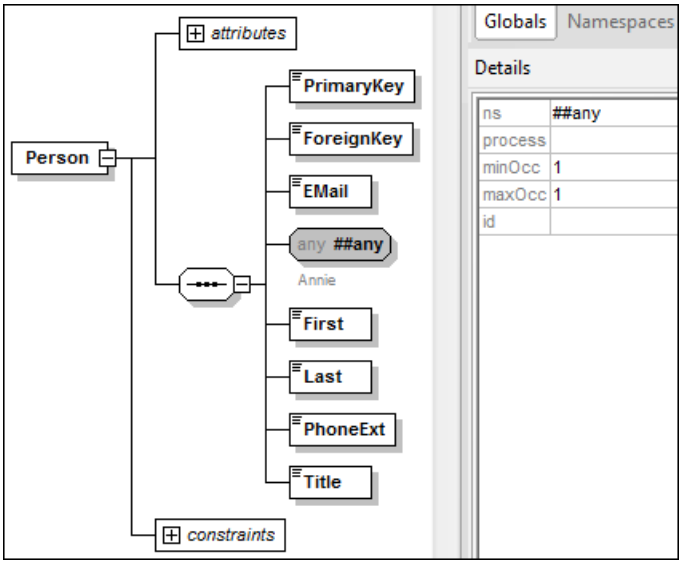


出力」ボタンをクリックすることにより、マークアップされたテキストを含むCDATA セクションが表示されます。

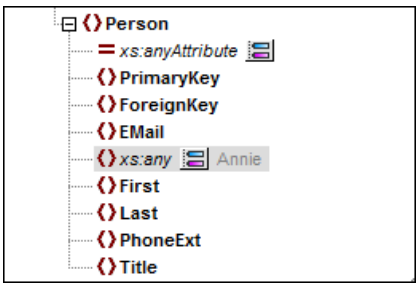
```
7     <Info>
8       <Title>MapForce</Title>
9       <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i>, <i>Java</i>, <i>C++</i> and <i>C#</i>. It is the definitive tool for
data integration and information leverage.]]></Description>
10    </Info>
```

7.1.9 ワイルドカード - xs:any / xs:anyAttribute

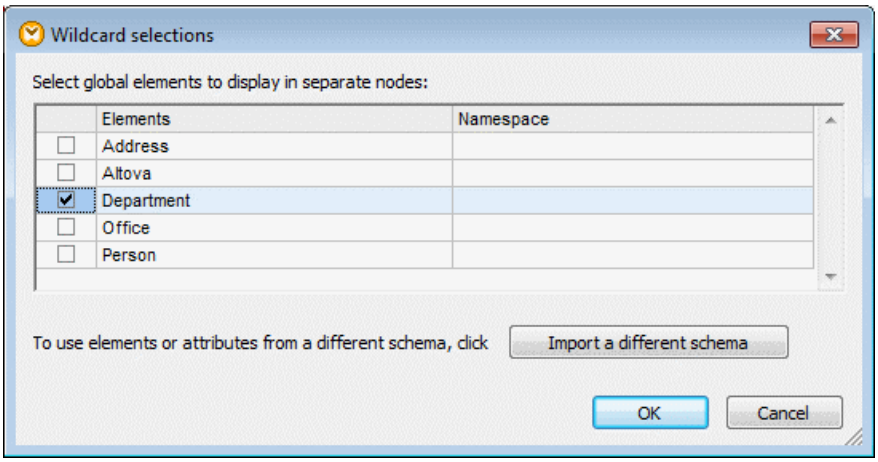
ワイルドカードxs:any (およびxs:anyAttribute) は、すべての要素/属性をスキーマから使用することを許可します。スクリーンショットはXMLSpy のスキーマビュー内の "any" 要素を表示しています。



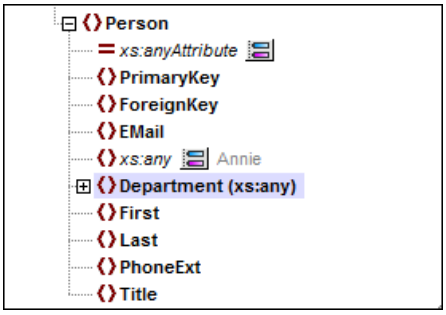
MapForce 内では、選択の変更 (excel1-compicon) ボタンがxs:any (またはxs:anyAttribute)要素の横に表示されます。



クリックすると、選択の変更 ボタンexcel1-compicon が開かれ、"ワイルドカードの選択 "ダイアログボックスが開かれます。エントリは、現在のスキーマ内で宣言されているグローバルな要素と属性を表示します。



クリックすると、選択の変更 ボタンexcel1-compicon が開かれ、"ワイルドカードの選択 "ダイアログボックスが開かれます。エントリは、現在のスキーマ内で宣言されているグローバルな要素と属性を表示します。



. これらのノードから、または、ノードへ、他の要素と同様にマップすることができます。

コンポーネント上では、ワイルドカード要素、または、属性は、名前に追加されている**(xs:any)** テキストとして識別されることができます。

ワイルドカード要素を削除するには、選択の変更 (excel1-compicon) ボタンをクリックして、"ワイルドカードの選択 "ダイアログボックスから選択の解除を行います。

ワイルドカードと動的なノード名

ワイルドカードから、または、ワイルドカードへのマッピングは、通常、使用することのできる、要素または属性がインスタンス内に表示されており、XML インスタンスがコンポーネントの XML スキーマにより宣言されている場合 (または、外部のスキーマからインポートされている場合) に適しています。しかしながら、インスタンス内に表示される要素または属性が多すぎ、スキーマ内で宣言されるには多すぎる場合があります。<message> の子要素の数が任意である次のインスタンスの場合を考慮してみてください:

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
   <line1>1</line1>
   <line2>2</line2>
   <line3>3</line3>
   ...............
   <line999></line999>
</message>
```

上記のようなシチュエーションでは、ノード名への動的なアクセスがワイルドカードの使用よりも適しています (ノード名のマッピング を参照)。

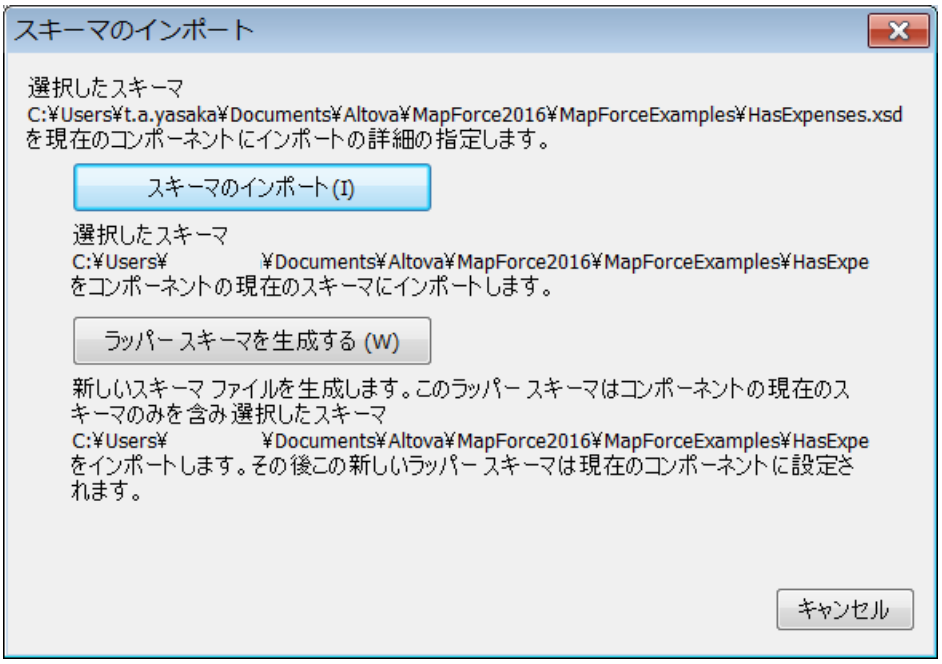
異なるスキーマからワイルドカードとして要素を追加する

コンポーネントに割り当てられた要素以外のスキーマからの要素は、ワイルドカードとして使用することができます。このような要素をコンポーネント上で表示するには、"ワイルドカードの選択 "ダイアログボックス上の**異なるスキーマのインポート**ボタンをクリックします。これにより以下の2つのオプションを持つ新規ダイアログボックスが開かれます:

1. スキーマのインポート

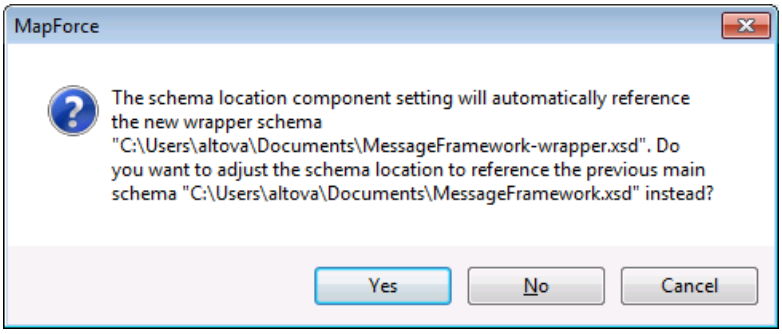
2. ラッパースキーマの生成

例えば、下のイメージは、**HasExpenses.xsd** という外部スキーマを現在のスキーマにインポートしようと試みると何が起こるかを表しています。



スキーマのインポート オプションは外部スキーマをコンポーネントに割り当てられている現在のスキーマにインポートします。このオプションはディスク上のコンポーネントの既存のスキーマをオーバーライドすることに注意してください。現在のスキーマがディスクではなくURL により開かれたリモートのスキーマである場合、変更することはできません (URL からコンポーネントを追加するを参照してください)。この場合、ラッパースキーマの生成オプションを使用してください。

ラッパースキーマの生成オプションは "ラッパー" スキーマという名前の新しいスキーマファイルを作成します。このオプションの利点は、コンテンツの既存のスキーマが変更されない点です。代わりに、既存のスキーマとインポートされるスキーマの両方を含む(ラッパースキーマという)新しいスキーマが作成されます。このオプションを選択すると、ラッパースキーマを保存する場所を問われます。デフォルトでは、ラッパースキーマは somefile-wrapper.xsd. という書式の名前を持ちます。ラッパースキーマを保存した後、デフォルトでコンポーネントに割り当てられ、ダイアログボックスがプロンプトします :



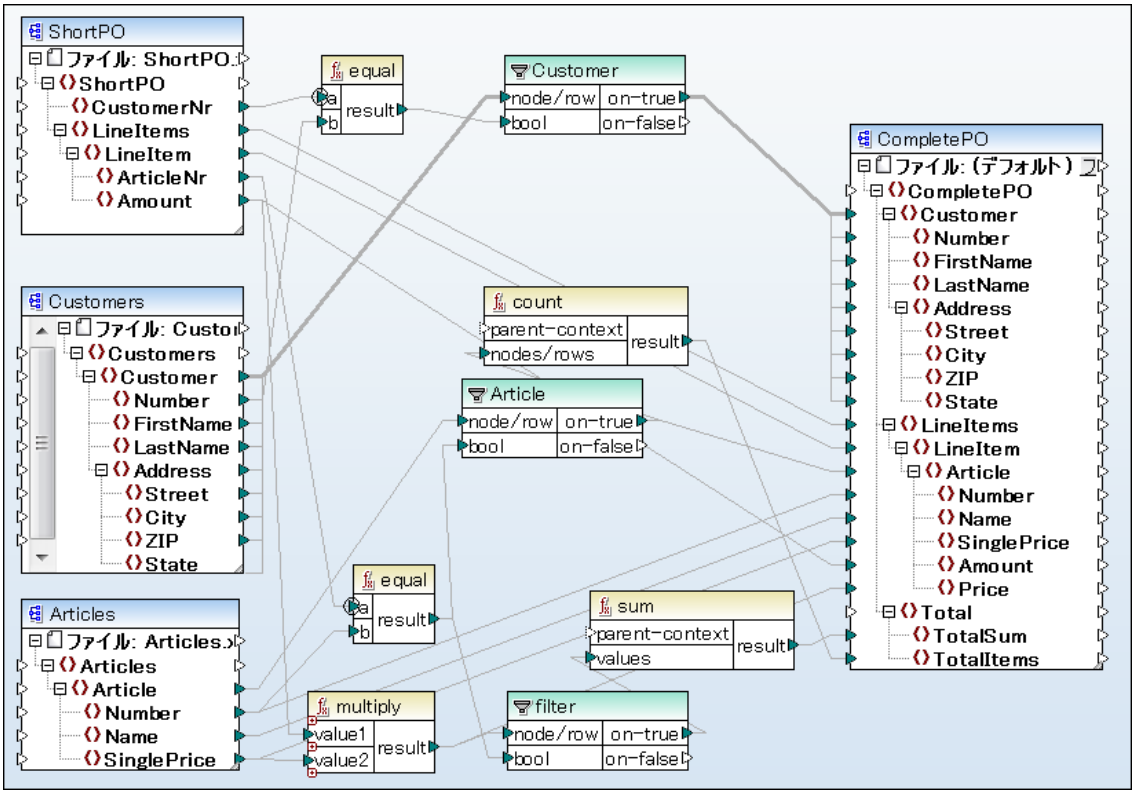
【はい】 をクリックして、前のスキーマに戻します。それ以外の場合は、**【いいえ】** をクリックして新規に作成された、コンポーネントに割り当てられたラッパースキーマを保持するためにいいえをクリックします。

7.1.10 複数のスキーマからのデータのマージ

MapForce では複数のファイルを 1つのターゲットファイルへマージすることができます。

以下にあるサンプルでは、複数のソースコンポーネント (複数のスキーマ) から得られたデータをターゲットスキーマにマージします。同一のスキーマをベースにした複数のファイルをマージする方法については、複数の入力または出力ファイルを動的に処理を参照ください。

...\MapForceExamples フォルダーにある **CompletePO.mfd** ファイルでは、 3つのファイルを 1つの XML ファイルへマージする方法を確認することができます



複数のソースコンポーネントデータが 1つのターゲットXML ファイル (CompletePO) へ組み合わされているという点に注目してください。

- **ShortPO** はXML インスタンスファイルに関連付けられたスキーマで、顧客番号と品物に関するデータ (LineItem ならびにAmount) だけが収められています。(このファイルには顧客番号が 3の顧客一人だけが収められています。)
- **Customers** はXML インスタンスファイルに関連付けられたスキーマで、顧客番号と、Name やAddress といった顧客に関する詳細情報が収められています。
- **Articles** はXML インスタンスファイルが関連付けられたスキーマで、品物名やその数、単価といった品物に関する情報が収められています。
- **CompletePO** は、 3つのXML インスタンスファイルから得られたデータを収めるための、インスタンスファイルを持たないスキーマファイルとなります。このファイルの階層構造により、XML データのマージと出力を行うことが可能になります。

このスキーマファイルはXMLSpy のようなXML エディターにより作成されており、MapForce から生成されたものではありません (CompletePO.xml インスタンスファイルが存在する場合、スキーマファイルの生成を行うことも可能です)。

CompletePO の構造はソースとなるXML ファイルの構造を組み合わせたものとなります。

フィルターコンポーネント (Customer) を使用することで、ShortPO とCustomer XML ファイルで顧客番号が一致するデータを検索することができ、関連するデータをターゲットのCompletePO コンポーネントへ渡すことができます。

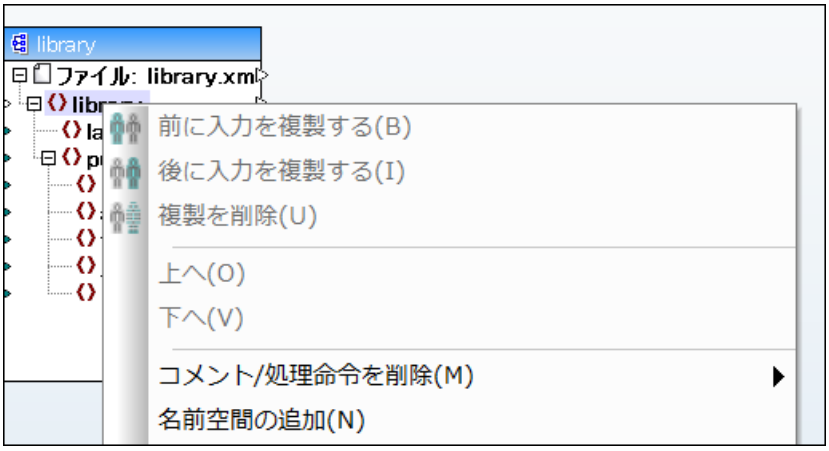
- "equal" 関数により、ShortPO の **CustomerNr** が Customer 以下にある **Number** と比較されます。
- ShortPO には 1つの顧客情報 (番号が 3) しか含まれていないため、顧客番号が 3となっている顧客の情報だけがフィルターコンポーネントには渡されます。
- フィルターコンポーネントの **node/ row** パラメーターは、bool パラメーターが true の時に **Customer** データを "bn- true" ノードへ渡します。つまり、顧客番号が 3の時だけ出力が行われます。
- 品物データは、他にある 2つのフィルターコンポーネントによりターゲットスキーマへ渡されます。

7.1.11 カスタム名前空間の宣言

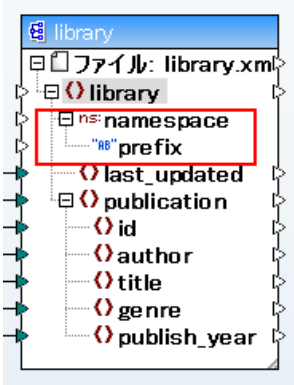
デフォルトでは、マッピングがXML 出力を生成する際に、各要素と属性の名前空間 (または、名前空間のセット) が自動的にMapForce によりターゲットコンポーネントと関連するスキーマから生成されます。これは、MapForce 内のデフォルトの振る舞いで、XML 出力の生成を含むマッピングのシナリオの大半に適切です。

しかしながら、結果のXML 出力内で要素の名前空間の管理を更に行う場合、マッピングから要素の名前空間を手動で宣言することが必要な可能性があります。

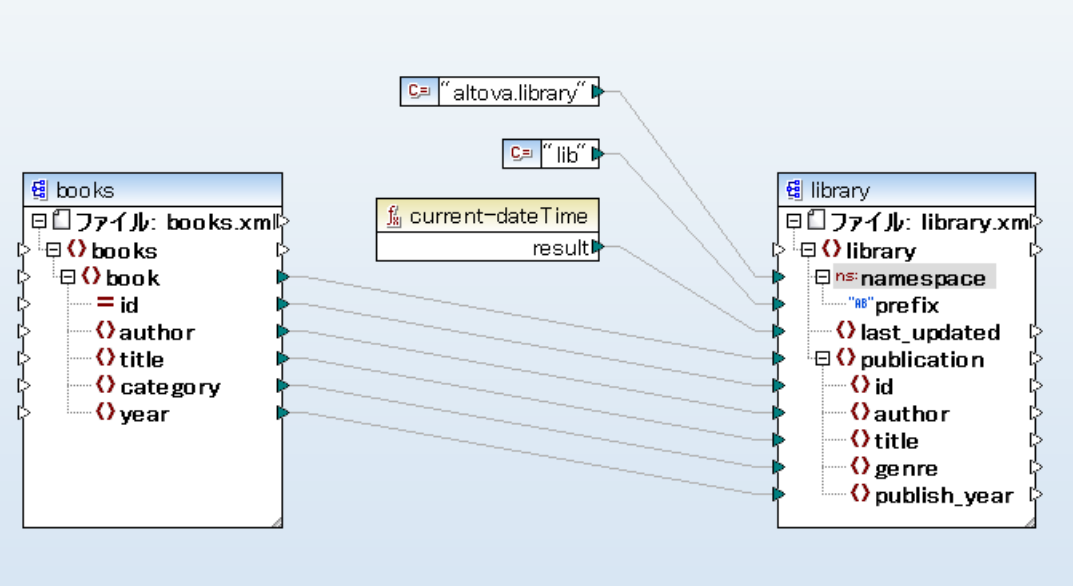
この仕組みを理解するには、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** 内にある**BooksToLibrary.mfd** マッピングを開きます。`library` ノードを右クリックし、コンテキストメニューから **「名前空間の追加」** を選択します。



`library` ノードの下で次の2つの新しいノードが使用することができるようになっていることに注意してください: `namespace` と`prefix`.



マッピングからの文字列の値をマップすることができます。下のイメージでは、名前空間「altova.library」とprefix「lib」を与える、2つの定数が(**挿入 | 定数**」メニューコマンドを使用して定義されています:



生成された出力内の結果としては、`xmlns:<prefix>="<namespace>"` 属性が、`<prefix>` と `<namespace>` がマッピング(この場合は定数)から来る値である箇所の要素に追加されます。生成された出力は、以下のようになります(ハイライトされている部分に注意してください):

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns:lib="altova.library" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:noNamespaceSchemaLocation="library.xsd">
...
```

メモ: カスタム名前空間の宣言(と**名前空間の追加**」コマンド)は、ターゲットXML コンポーネントに取り有益であり、要素のみに適用することができます。属性に対して、**名前空間の追加**」コマンドは、属性とワイルドカードノードに対しては使用することができません。ノードの子への動的なアクセスが有効化されている場合でも、または、**全てをコピー**」接続を使用してノードがデータを受け取るノードの場合でも使用することはできません(ノード名のマッピングを参照してください)。

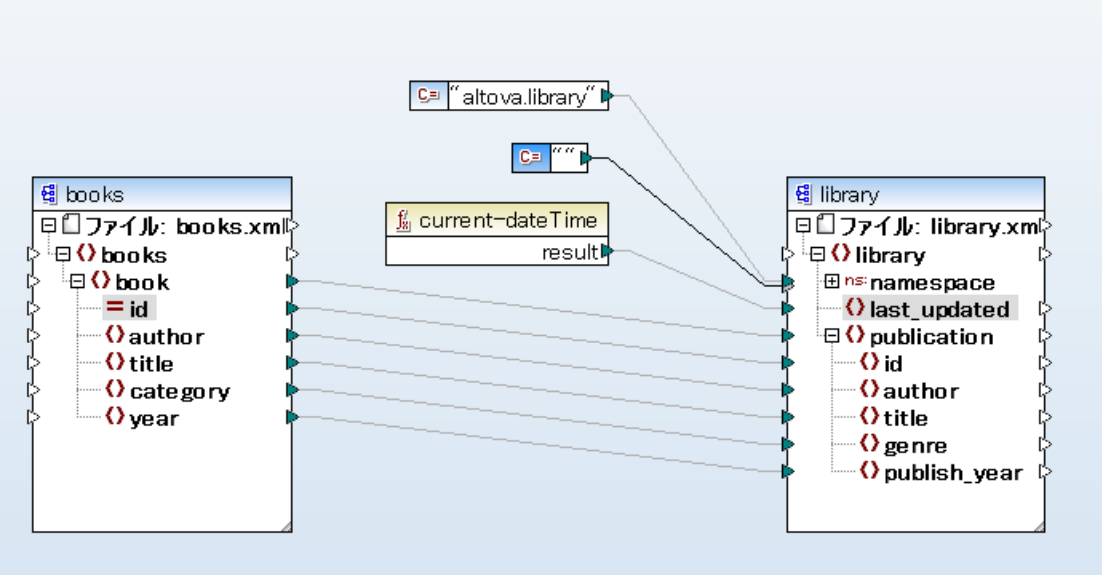
必要に応じて、複数の名前空間を同じ要素のために宣言することができます。これを行うには、ノードをもう一度右クリックし、コンテキストメニューから**名前空間の追加**」を選択します。新規のプレフィックスと名前空間の値に接続することので

きる名前空間とプレフィックスノードの新規のペアを使用できるようになります。

以前に追加された名前空間の宣言を削除するには、 ns:namespace ノードを右クリックし、コンテキストメニューから **「名前空間の削除」**を選択します。

> 空の値が与えられる場合でも、namespace とprefix 入力コネクタの両方がマップされている必要があります。

(xmlns="mydefaultnamespace"のフォーマットの) デフォルトの名前空間を宣言する場合、prefix に対して空の文字列をマップします。このケースをマップ上で確認する場合は、2番目の定数が空の文字列になるように上のサンプルマッピングを編集します。



結果の出力は以下のようになります :

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns="altova.library" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="library.xsd">
...
```

属性の名前のためにプレフィックスを作成する場合、例えば、<number prod:id="prod557">557</number> 、ノードの属性への動的なアクセスを有効化する、または<number> のためにprod:id 属性を持つようにスキーマを編集することにより属性の名前のためにプレフィックスを作成することができます (次を参照してください: ノード名のマッピング)。

7.1.12 デジタル署名

XML デジタル署名はXML ドキュメントを暗号化されたコードにより署名するためのW3C 仕様で、XML ドキュメントが変更されていないことを保証するために使用することができます。MapForce のXML 署名機能ではUSA-SHA1 ならびにDSA-SHA1 型の証明書がサポートされます。

XML 署名に関する詳細については、XML 署名に関するW3C の仕様 http://www.w3.org/TR/xmldsig-core/ を参照ください。

MapForce ではXML ならびにXBRL 出力ファイルに対してXML デジタル署名の作成がサポートされます。デジタル署

名は出力ターゲットがビルトインとなっている場合のみ生成することができます。出力ボタンがクリックされた時、また出力ファイルが保存された時に、生成された結果ファイルに対して署名が作成されます。

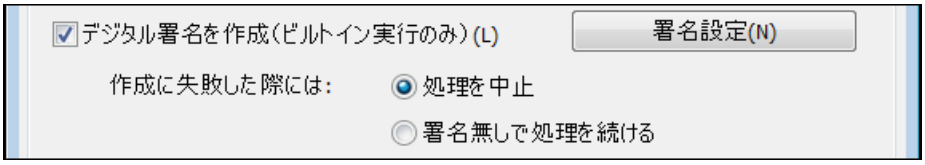
メモ:MapForce Server はデジタル署名をサポートしません。

デジタル署名は出力ドキュメント最後の要素として埋め込むことができるほか、独立した署名ファイルとしても保管することができます。

- "Enveloped" が選択されている場合、XML ファイルにあるルート要素直下の最後の子要素として署名が作成されます。
- "Detached" が選択されている場合、独立したドキュメントとして署名ファイルが生成されます。

デジタル署名の生成を有効にするには:

1. 出力コンポーネントのヘッダーをダブルクリックするか、「コンポーネント| プロパティ」を選択することで表示される、出力コンポーネントのコンポーネント設定ダイアログボックスを開きます。
2. デジタル署名を作成チェックボックスを有効にします。



3. 必要な設定を入力するXML 署名設定ダイアログボックスが開かれます。 (XML 署名設定を参照ください)

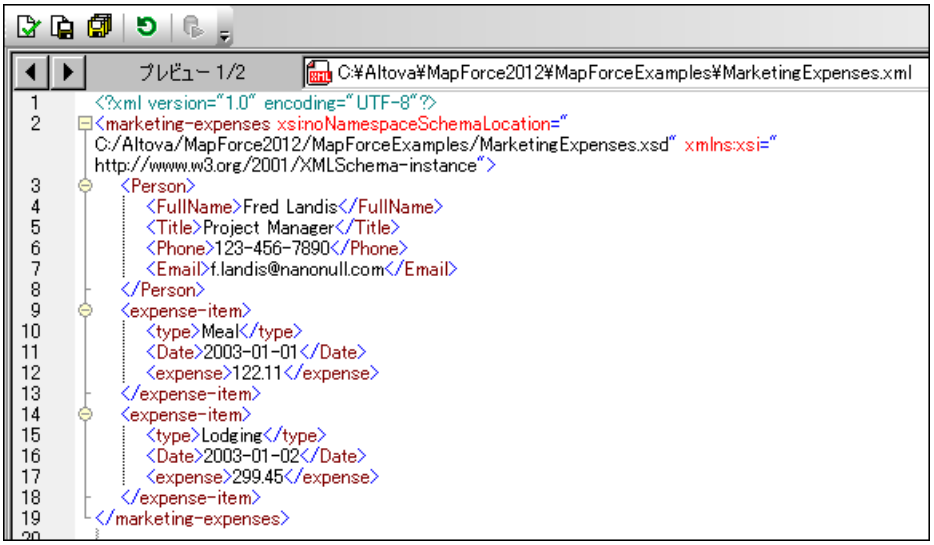
デジタル署名の設定を変更する:

1. 出力コンポーネントのヘッダーをダブルクリックすることで表示されるコンポーネント設定ダイアログボックスを開きます。
2. 署名設定ボタンをクリックして、XML 署名設定 ダイアログボックスを開きます。
3. 設定を入力して **「OK」** をクリックします。

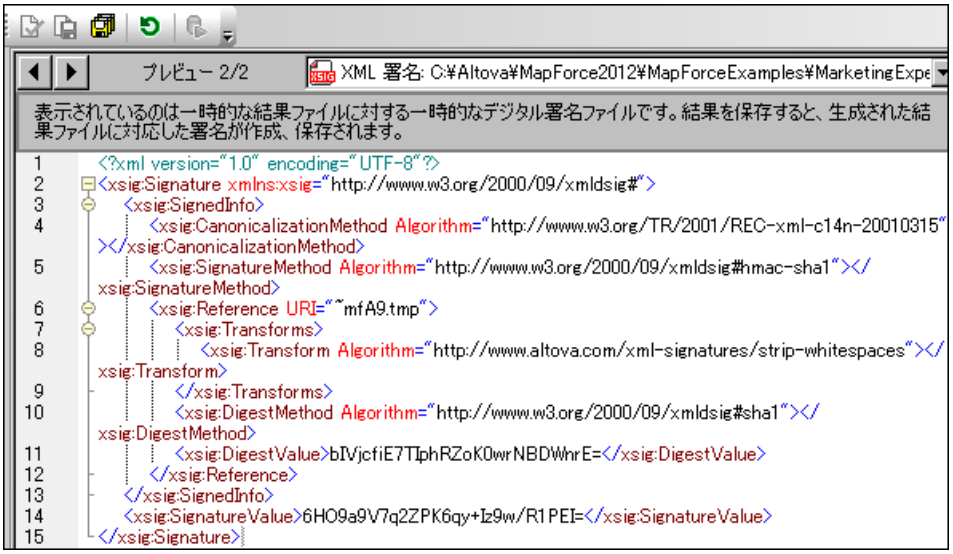
以下の例では **..\MapForceExamples** フォルダー以下の **MarketingExpress_DetachedSignature.mfd** ファイルを使用します:

1. MarketingExpenses ターゲットコンポーネントをダブルクリックして、署名設定ボタンをクリックします。
2. 上のスクリーンショットにあるような選択を行い、 **「OK」** をクリックしてダイアログボックスを閉じます。
3. 出力ボタンをクリックして、マッピングの結果を確認します。

   2 つのファイルがプレビューウィンドウに生成されました。**MarketingExpenses.xml** がターゲットコンポーネントの結果ファイルとなります。.

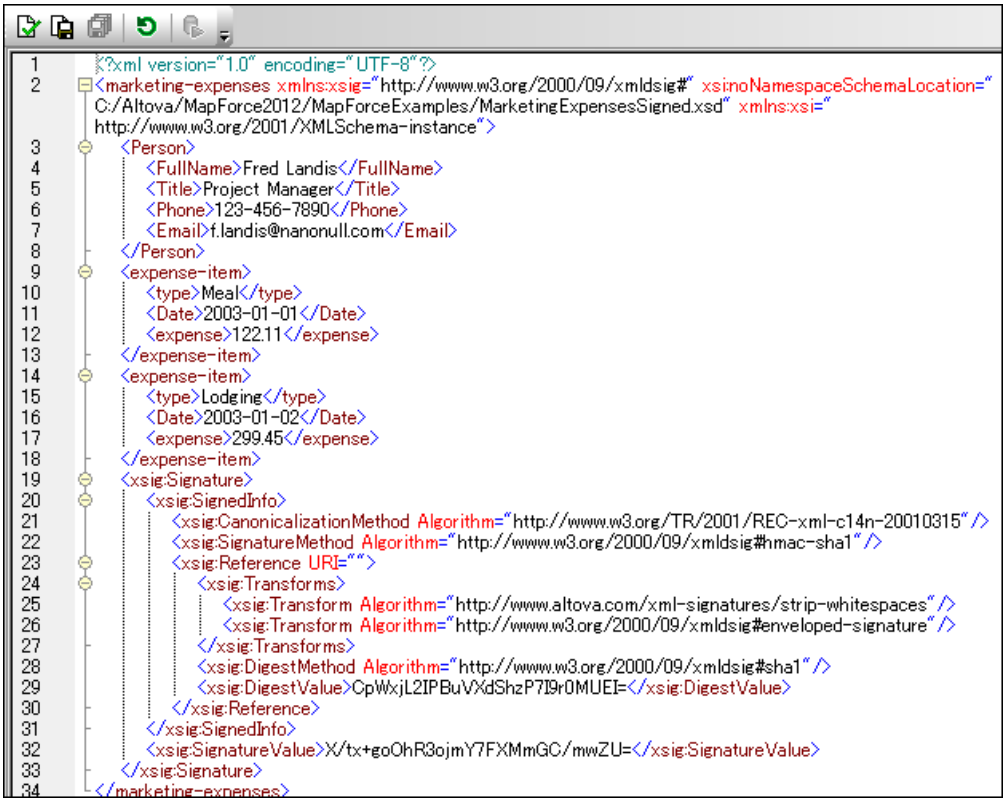


以下に示される **MarketExpress.xml.xsig** はターゲットコンポーネントにより生成された一時的なデジタル署名ファイルです。



出力プレビューツールバーにある**生成された出力を全て保存**アイコン をクリックします。
これで出力ディレクトリに XML ならびに XSIG ファイルが生成されます。

..\MapForceExamples フォルダー内に収められている**MarketingExpenses_EnvelopedSignature.mfd** ファイルでは、"Enveloped" 設定で生成される署名の出力を確認することができます。



```
<?xml version="1.0" encoding="UTF-8"?>
<marketing-expenses xmlns:xsig="http://www.w3.org/2000/09/xmldsig#" xsi:noNamespaceSchemaLocation="
C:/Altova/MapForce2012/MapForceExamples/MarketingExpensesSigned.xsd" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
   <Person>
      <FullName>Fred Landis</FullName>
      <Title>Project Manager</Title>
      <Phone>123-456-7890</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <xsig:Signature>
      <xsig:SignedInfo>
         <xsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
         <xsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1" />
         <xsig:Reference URI="">
            <xsig:Transforms>
               <xsig:Transform Algorithm="http://www.altova.com/xml-signatures/strip-whitespaces" />
               <xsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
            </xsig:Transforms>
            <xsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <xsig:DigestValue>CpWxjL2IPBuVXdShzP7I9r0MUEI=</xsig:DigestValue>
         </xsig:Reference>
      </xsig:SignedInfo>
      <xsig:SignatureValue>X/tx+goOhR3ojmY7FXMmGC/mwZU=</xsig:SignatureValue>
   </xsig:Signature>
</marketing-expenses>
```

XML ドキュメントの妥当性

XML 署名が XML ドキュメント内に埋め込まれている場合、http://www.w3.org/2000/09/xmlsig# 名前空間内の Signature 要素が XML ドキュメントに追加されます。スキーマに従うかたちでドキュメントの妥当性を保つには、適切な要素宣言をスキーマ内に含める必要があります。MapForce では Enveloped オプションを使用した署名の埋込みが行われます：

- *Enveloped 要素がルート (またはドキュメント) 要素以下にある最後の子要素として、Signature 要素が作成されます。*

  XML ドキュメントのスキーマに変更を加えたくない場合、"Detached" オプションを使用することで XML 署名を独立した外部ファイルとして作成することができます。

Enveloped 署名の Signature 要素を許可するために使用される XML スキーマの抜粋を以下に示します。これらのサンプルをガイドとして、使用中のスキーマを修正することができます。

2 つのうちの最初のサンプルでは、XML 署名スキーマがユーザーのスキーマにインポートされます。XML 署名に使用されるスキーマは、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>
```

以下にある) 2番目のオプションでは、他の名前空間にある要素全てにマッチするような、全般的なワイルドカード要素が追加されます。processContents 属性を lax にセットすることで、検証の際にこの要素をスキップするようになります (マッチする要素宣言が見つからないため)。結果として、ユーザーは XML 署名スキーマを参照する必要がなくなります。このオプションの欠点は、XML ドキュメントの妥当性を損なうこと無く、(Signature 要素以外の) 全ての要素を特定の場所に追加することができてしまうという点です。

```
<?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>
```

7.1.12.1 XML 署名設定

署名に関する設定はコンポーネント設定ダイアログボックスにて、個別のコンポーネントに対して保管することができるとともに、MFD ファイルが保存された際に全ての設定が保存されます。



XML 署名設定ダイアログボックス

認証方法：証明書とパスワード

署名は証明書またはパスワードをベースに作成することができます。使用する方法のラジオボタンを選択してください：

- *証明書 ：*
  証明書を使用するには、証明書に秘密鍵が含まれており、アクセスすることができる証明書ストアに証明書が配置されている必要があります。署名は証明書の秘密鍵により生成されます。署名の検証を行うには、証明書（又はその公開鍵バージョン）へアクセスする必要があります。証明書の公開鍵は署名の検証を行うために使用されます。秘密鍵ならびに公開鍵の証明書を選択するには、選択ボタンをクリックして証明書を参照してください。

- *パスワード：*
  5 文字から16文字の長さのパスワードを入力します。このパスワードにより署名が作成され、署名の検証を行うために必要になります。ダイアログボックスの **「OK」** ボタンは、パスワードの必須要件が満たされたときに有効になります。

- *MFD ファイルにパスワードを保存 ：*
  有効になっている場合、パスワードフィールドに入力されたパスワードが、人間には読むことのできない難読化されたかたちで MFD ファイル内部に保存されます。このパスワードを使用することで、MFD ファイルへアクセスすることができる人ならば誰でも署名を作成することができるようになります。

変換

XML データが変換され、その結果が署名の作成に使用されます。正規化アルゴリズムを指定することで、署名の計算を行う前に、そのアルゴリズムがファイルの XML データ (SignedInfo コンテンツ) へ適用されます。アルゴリズム間における重要な違いを以下に記します：

- *コメント付きのCanonical XML：*
  コメントが署名の計算に含まれる場合、XML データのコメントに変更を加えることで、検証に失敗するようになります。コメントを署名の計算に含まない場合、ドキュメントの署名が行われた後でも、コメントの追加や変更を行うことで署名が失敗することはありません

  メモ：
  「コメント付き..」は "Detached" が選択されている場合のみ選択可能になります。

- *Base64:*
  XML ドキュメントのルート (またはドキュメント) 要素は Base64 によりエンコードされていると認識され、バイナリ形式で読み取られます。ルート要素が Base64 でない場合、要素の型により、エラーが返されるか空の要素として読み取られます。

- *無し：*
  変換は行われず、ディスクに保存されているバイナリファイルから読み取られた XML データが、署名の作成に直接使用されます。署名作成後に行われた変更により、署名の検証が失敗するようになります。

  しかし、**XML 要素間の空白スペースを取り去る**チェックボックスにチェックが入っている場合、全ての空白スペースが取り除かれ、空白スペースに対する変更が無視されるようになります。

  無しオプションとCanonicalization オプションの主な違いは、正規化によりXML データストリームが作成され、属性の順序といった違いが正規化されるという点にあります。結果として Canonicalization 変換では属性の順序変更といったものも正規化され検証が成功する一方、変換を行わない場合そのような変更も検証の結果に反映されます (つまり検証は失敗します)。

  デフォルトの正規化は署名が埋め込まれている (Enveloped) 場合だけ行われることに注意してください。そのため、署名ファイルが独立している場合 (Detached)、無しが選択されている場合、または空白スペースを取り去るチェックボックスにチェックが入っていない場合、XML データがそのまま使用されます。

署名の配置

署名は XML ファイルの中に配置することができるほか、独立したファイルとしても作成することができます。以下のオプションが利用可能です：

- *Enveloped:*
  署名要素がルート (ドキュメント) 要素直下にある最後の子要素として作成されます。出力 XML を妥当とするためには、関連する XML スキーマに署名の定義を行う要素を含める必要があります。詳細は、このセクションの上部を参照ください。

- *Detached:*
  XML 署名を独立したファイルとして作成します。この場合、署名ファイルのファイル拡張子を指定し、(i) XML ファイルに拡張子を追加する (例：test.xml.xsig) か、(i) 拡張子で XML ファイルの XML 拡張子を置き換える (例：test.xsig) かを選択することができます。署名ファイル内部で指定される XML ファイルへのパスを相対パスとするか、絶対パスとするかも選択することができます。

メモ： XML スキーマ (xsd) ファイルならびに XBRL ファイルに対する XML 署名は外部署名ファイルとしてしか作成することができません。WSDL ファイルの場合、外部ファイルならびに WSDL ファイル内部へ署名を埋め込む (enveloped) ことができます。

メモ： XML 署名が独立したファイルとして作成された場合、署名ファイル内部で行われる参照により XML ファイルと署名ファイルの関連付けが行われます。そのため、署名が外部ファイルに収められている状態の検証は、XML ファイルがアクティブな状態ではなく、署名ファイルがアクティブな状態で行われる必要があります。

鍵情報を追加

署名が証明書をベースとしている場合、鍵情報を追加 オプションを選択することができるようになります。

鍵情報を追加オプションが有効になっている場合、公開鍵情報が署名内部に配置され、有効になっていない場合、鍵情報は署名に含められません。鍵情報を追加することにより、証明書 ならびにその中に含まれる公開鍵情報) は、検証処理にて必要となりません 鍵情報は既に署名内に収められているため)。

署名設定 - 無効の場合

署名設定が無効の場合、MapForce はデジタル署名を作成することができません。以下の場合、署名設定が無効となります：

- 選択された証明書へアクセスすることができないか、XML ドキュメントの署名に使用することができない、または
- 「MFD ファイルにパスワードを保存」オプションにチェックされておらず、パスワードが設定されていない

出力ボタンをクリックすると、以下のメッセージが表示されます
パスワードの場合：

> コンポーネント "MarketingExpressSigned" の出力を署名するためのパスワードを指定してください

証明書の場合：

> コンポーネント "MarketingExpressSigned" の出力を署名するために使用する証明書が含まれているストアを選択してください

パスワードまたは証明書が選択されていない場合、処理が停止するか、署名なしで出力が行われます。この振る舞いはコンポーネント設定ダイアログボックスの処理を中止または署名なしで処理を続けるラジオボタンにより指定することができます。

コマンドラインからマッピングが実行された場合、ダイアログボックスは表示されません。マッピングの処理は中止されるか、署名無しで継続されることになります。

7.2 MapForce とデータベース

Altova Web サイト データベースマッピング

MapForce 2017 ではデータベースから、XML、フラットファイル、と他のデータベースフォーマットへのマッピングがサポートされています。MapForce Enterprise エディションでは、追加してデータベースをEDI フォーマット、Excel 2007+、JSON、XBRL とWeb サービスへのマッピングがサポートされます。

次のデータベースはサポートされています。各データベースの使用できるルート オブジェクトもリストされています。Altova は他のODBC/ADO データベースへのサポートの向上のための努力をしていますが、以下にリストされたデータベースのみで接続とデータ処理のテストに成功しています。使用中のAltova アプリケーションが、64 ビット バージョンの場合、接続されているデータベース特定の 64 ビットデータベース ドライバーにアクセスしてください。Altova アプリケーションはIBM iSeries データベースのロジカル ファイルをサポートし、ロジカル ファイルを表示します。

データベース	ルートオブジェクト	メモ
Firebird 2.5.4	データベース	
IBM DB2 8.x, 9.1, 9.5, 9.7, 10.1, 10.5	スキーマ	
IBM DB2 for i 6.1, 7.1	スキーマ	ロジカルファイルはサポートされ、ビューとして表示されています。
IBM Informix 11.70	データベース	Informix はADO、JDBC およびODBC を介した接続をサポートします。実装は、すべてのコード生成言語による大きなオブジェクトデータ型をサポートしません。MapForce は、(コード生成中) これらのデータ型が使用された場合エラーメッセージを表示します。
Microsoft Access 2003, 2007, 2010, 2013	データベース	
Microsoft Azure SQL Database	データベース	SQL Server 2016 codebase
Microsoft SQL Server 2005, 2008, 2012, 2014, 2016	データベース	
MySQL 5.0, 5.1, 5.5, 5.6	データベース	
Oracle 9i, 10g, 11g, 12c	スキーマ	
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1, 9.4, 9.5	データベース	PostgreSQL 接続は、ODBC またはJDBC などのインターフェイス (ドライバ) によりネイティブとドライバーベースの接続でサポートされています。ネイティブな接続はドライバを必要としません。
Progress OpenEdge 11.6	データベース	
SQLite 3.x	データベース	SQLite 接続は、ネイティブとしてサポートされます。SQLite データベースファイルへの直接の接続です。別途のドライバーは必要ありません。
Sybase ASE15	データベース	

多種の実行環境内でのデータベース マッピング

マッピングからプログラムコードを生成する場合、または、MapForce Server 実行可能ファイルにマッピングをコンパイルする場合、生成されたファイルと共に保存されたデータベース接続の詳細は、対応するドライバにより適用、または、選択されたターゲット環境によりサポートされています。 例えば、マッピング変換言語がJava に設定されている場合、ADO 接続は、マッピングからJava コードが生成されると、JDBC に変換されます。

MapForce 以外の環境でマッピングが実行されると、データベース接続の詳細がマッピングを実行するマシンに適合していることを確認してください (例えば、データベースドライバがインストールされている、データベースのパスが正確である、データベースサーバーへのアクセスがあるなど)。

データベースへの接続の種類の一部は、下のテーブルで示されているように、ターゲット環境の一部でサポートされていません。

接続の型 実行環境	C#	C++	Java	Windows 上の MapForce Server	Linux/Mac 上の MapForce Server
ADO	ADO ブリッジ	現状のまま	JDBC に変換されます	現状のまま	JDBC に変換されます
ADO.NET	現状のまま	ユーザー定義	JDBC に変換されます	現状のまま	JDBC に変換されます
JDBC	ユーザー定義	ユーザー定義	現状のまま	現状のまま	現状のまま
ODBC	ODBC ブリッジ	ODBC ブリッジ	JDBC に変換されます	現状のまま	JDBC に変換されます
ネイティブ PostgreSQL	サポート無し	サポート無し	サポート無し	現状のまま	現状のまま
ネイティブ SQLite	サポート無し	サポート無し	サポート無し	現状のまま	現状のまま

テーブル内のレジェンド:

- "現状のまま"は、データベース接続 型 (例えば、JDBC、ODBC) が、MapForce で定義されたままという意味です。
- "JDBC に変換されます"は、データベース接続が、JDBC のようなデータベース接続 URL に変換されるという意味です。
- "ADO ブリッジ"、または、"ODBC ブリッジ"は、接続文字列は、MapForce 内で定義されたままですが、生成されるコードは、ADO ブリッジ (、または、ODBC ブリッジ) としての役割を果たす適切なクラスを使用するという意味です。例えば、**System.Data.OleDb.OleDbConnection** 、または、**System.Data.Odbc.OdbcConnection** 等があげられます。
- "ユーザー定義"は、生成されたコード内で接続が作動するためには、手動で接続の情報をデータベースコンポーネント設定 ダイアログボックスに入力するという意味です。場合により異なりますが、接続の詳細は **ADO/OLE-DB-specific settings** 、または、 **JDBC-specific settings**.に入力される必要があります。

次も参照してください:

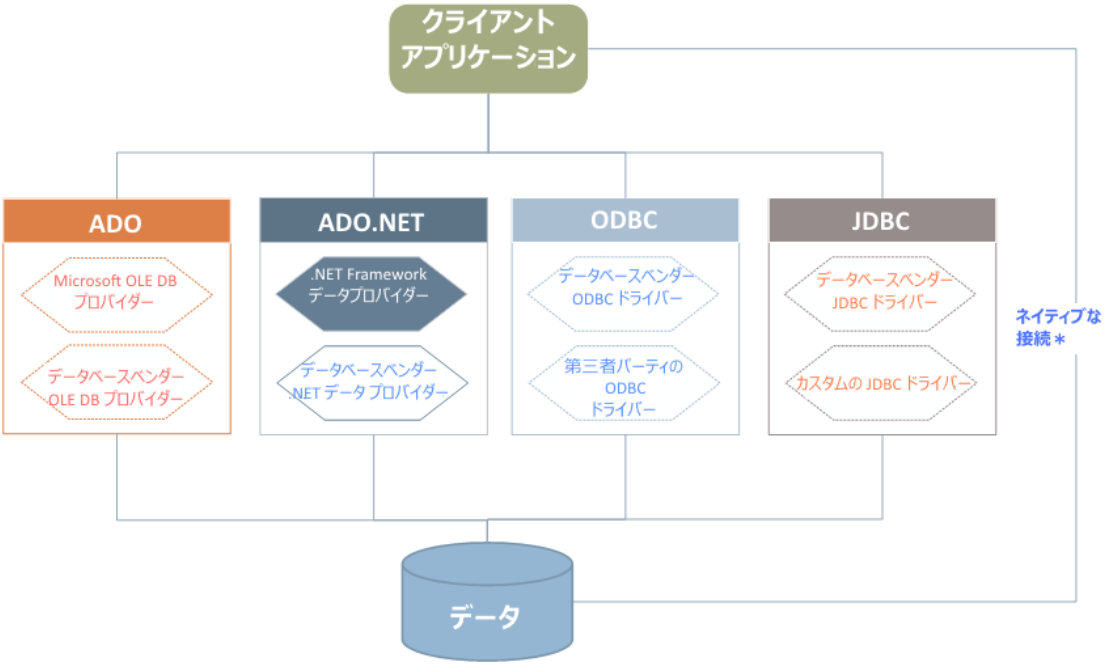
- Linux とMac でのデータベース接続
- マッピングをMapForce Server 実行可能ファイルにコンパイルする
- コードジェネレーター

7.2.1 データベースへの接続

最も簡単なケースとして、データベースは、Microsoft Access、または、SQLite データベースファイルなどのローカルファイルが挙げられます。高度なケースの例としては、データベースは、アプリケーションが接続し、データを消費するオペレーティングシステムと同じオペレーティングシステムを使用するとは限らない、リモート、または、ネットワークデータベースサーバーに存在します。例えば、MapForce が、Windows で作動し、データ (例えば MySQL) へアクセスするデータベースが Linux マシーンで作動することも可能です。

多種のデータベース型とリモートおよびローカルで作業するには、MapForce は、データ接続インターフェイスとオペレーティングシステム内で使用することのできる、または、定期的にメジャーなデータベースベンダーによりリリースされるデータベースドライバーに従います。常に進化するデータベース技術の景観の中、このアプローチは、クロスプラットフォームに対する柔軟性および相互運用性に対応します。すなわち、MapForce により、以下のデータアクセス技術により主要なデータベースにアクセスすることができます

次の図は、最も簡単な方法で、(ジェネリックなクライアントアプリケーションとして表示されている) MapForce と (データベースサーバー、または、データベースファイルであることができる) データ保管の間で使用することのできるデータ接続のオプションを示しています。



** 直接なネイティブな接続 は、SQLite とPostgreSQL データベースのためにサポートされています。使用中のシステムにインストールするために追加ドライバーは必要ではありません。*

上の図で示されるように、MapForce は、主なデータベース型に次のアクセス技術を使用してアクセスすることができます :

- 基となるOLE DB (Object Linking and Embedding, Database) プロバイダーを使用するADO (Microsoft® ActiveX® Data Objects)
- ADO.NET (データのインタアクションを有効化するMicrosoft .NET Framework 内で使用することのできるライブラリのセット)
- JDBC (Java データベース接続)
- ODBC (Open データベース接続)

選択するデータベースへの接続のインターフェイスは、既存のソフトウェアのインフラストラクチャによります。データのアクセス技術と、接続するデータベースと密に統合することのできる、データベースドライバーを通常選択します。例えば、Microsoft Access 2013 データベースに接続するには、**Microsoft Office Access データベース Engine OLE DB プロバイダー** などのネイティブなプロバイダを使用する ADO 接続文字列を作成します。に接続するには、他方、最新の JDBC、ODBC、または ADO.NET インターフェイスを Oracle Web サイトからダウンロードします。

Windows 製品（Microsoft Access または SQL Server など）のためのドライバーは、Windows オペレーティングシステムで使用可能ですが、他のデータベースの種類では使用することができない可能性があります。メジャーなデータベースベンダーは、対応するデータベースに ADO、ADO.NET、ODBC、または JDBC の組み合わせからクロスプラットフォームアクセスをすることができる、一般使用可能なデータベースクライアントソフトウェアとドライバーを定期的にリリースします。更に、上記の技術のためのサードパーティーのドライバーを使用することができます。多くの場合は、1つ以上の方法で必要なデータベースへ使用中のオペレーティングシステムから接続することができます、この結果、MapForce からも接続することができます。使用可能な機能、パフォーマンスパラメーター、と既知の問題はデータアクセス技術および使用されるドライバーにより異なります。

7.2.1.1 データベース接続ウィザードの開始

データベース接続を必要とするアクションには、接続をセットアップするために必要なステップを案内するウィザードが表示されます。

ウィザードのステップを開始する前に、データベースの一部の種類は、データベースドライバーまたはデータベースクライアントソフトウェアなどの複数のデータベースの必須条件を個別にインストールまたは構成する必要があることに注意してください。これらは、通常各データベースのベンダーにより提供され、特定の Windows バージョン用に調整されたドキュメントを含みます。データベースの種類別にグループ化されたデータベースドライバーのリストに関しては、を[データベースドライバーの概要](#)参照してください。

データベースをソースまたはコンポーネントをマッピングに追加するには以下を行ってください：

- **挿入** メニューから、**データベース**をクリックしてください。

データベースを再利用可能なグローバルリソースとして追加するには以下を行ってください：

1. **ツール** メニューから、**[グローバルリソース]**をクリックします。
2. **[追加]** をクリックして、データベースをクリックします。
3. **[データベースの選択]** をクリックします。



データベースの種類を選択した後、**「次へ」**をクリックします。スクリーンの指示はデータベースの種類（ADO、ODBC、JDBC）およびドライバーにより異なります。

例えば、各データベースの種類に適用できるサンプルはデータベース接続のサンプルのセクションを参照してください。各データベースへのアクセス技術に関しては以下のトピックを参照してください：

- ADO 接続の設定
- ADO.NET 接続の設定
- ODBC 接続の設定
- JDBC 接続の設定

7.2.1.2 データベースドライバー概要

以下のテーブル、特定のデータベースに特定のデータアクセス技術を使用して接続するためのはよく使用されるデータベースのドライバーのリストです。このリストは、完全または規範的なものではありません。以下に表示されているドライバに加え他のネイティブまたはサードパーティー代替を使用することができます。

(下のテーブル内でハイライトされている)Windows オペレーティングシステムのデフォルトの複数のデータベースドライバーを使用することができますが、代替ドライバーをダウンロードする必要がある可能性があります。データベースの一部に対して、オペレーティングシステムに付随するドライバーよりもデータベースベンダーより提供される最新のドライバーのパフォーマンスの方が良い場合があります。

一部の例外を除いて、データベースのベンダーは、通常必要なデータベースドライバーを含むデータベースクライアントソフトウェアを提供します。または、インストールするドライバーとコンポーネントのインストール中にオプションを与えます。データベースクライアントソフトウェアは、通常、データベースの管理と接続、およびデータベースクライアントとそのコンポーネントのインストールと構成を簡素化するために管理と構成ユーティリティにより構成されています。

データベースクライアントを正確に構成することは、データベースへの接続を成功させるために必須です。データベースをインストールして、使用する前に、各データベースバージョンおよび各 Windows バージョンにより通常異なるデータベースのインストールおよび構成の手順を確認することが奨励されます。

データベース	インターフェイス	ドライバー
Firebird	ADO.NET	Firebird ADO.NET データプロバイダー (http://www.firebirdsql.org/en/additional-downloads/)
	JDBC	Firebird JDBC ドライバー (http://www.firebirdsql.org/en/jdbc-driver/)
	ODBC	Firebird ODBC ドライバー (http://www.firebirdsql.org/en/odbc-driver/)
IBM DB2	ADO	DB2 のための IBM OLE DB プロバイダー
	ADO.NET	.NET のための IBM Data Server プロバイダー
	JDBC	JDBC とSQLJ のための IBM Data Server ドライバー
	ODBC	IBM DB2 ODBC ドライバー
IBM DB2 for i	ADO	• i5/OS IBMDA400 OLE DB プロバイダーのための IBM DB2 • i5/OS IBMDARLA OLE DB プロバイダーのための IBM DB2 • i5/OS IBMDASQL OLE DB プロバイダーのための IBM DB2
	ADO.NET	Windows .NET プロバイダーのための IBM i Access
	JDBC	Java JDBC ドライバーのための IBM Toolbox
	ODBC	iSeries Access ODBC ドライバー
IBM Informix	ADO	IBM Informix OLE DB プロバイダー
	JDBC	IBM Informix JDBC ドライバー
	ODBC	IBM Informix ODBC ドライバー
Microsoft Access	ADO	• **Microsoft Jet OLE DB プロバイダー** • **Microsoft Access Database Engine OLE DB プロバイダー**
	ADO.NET	**OLE DB のための .NET Framework データプロバイダー**
	ODBC	• **Microsoft Access ドライバー**
Microsoft SQL Server	ADO	• **SQL Server のための Microsoft OLE DB プロバイダー** • **SQL Server Native Client**
	ADO.NET	• **SQL Server のための .NET Framework データプロバイダー** • **OLE DB のための .NET Framework データプロバイダー**
	JDBC	• SQL Server のための Microsoft JDBC ドライバー (http://msdn.microsoft.com/en-us/data/aa937724.aspx)
	ODBC	• **SQL Server Native Client**
MySQL	ADO.NET	Connector/NET (http://dev.mysql.com/downloads/connector/net/)
	JDBC	Connector/J (http://dev.mysql.com/downloads/connector/j/)
	ODBC	Connector/ODBC (http://dev.mysql.com/downloads/connector/odbc/)

データベース	インターフェイス	ドライバー
Oracle	ADO	• OLE DB のための Oracle プロバイダー • **Oracle のための Microsoft OLE DB プロバイダー**
	ADO.NET	.NET のための Oracle データプロバイダー (http://www.oracle.com/technetwork/topics/dotnet/index-085163.html)
	JDBC	• JDBC Thin ドライバー • JDBC Oracle Call Interface (OCI) ドライバー これらのドライバーは、通常、データベースクライアントのインストール中にインストールされます。(Thin ドライバーではない) OCI ドライバーを使用する接続は、Oracle XML DB のコンポーネントを使用しています。
	ODBC	• **Oracle のための Microsoft ODBC** • Oracle ODBC ドライバー (通常、Oracle データベースクライアントのインストール中にインストールされます)。
PostgreSQL	JDBC	Postgre JDBC ドライバー (https://jdbc.postgresql.org/download.html)
	ODBC	psqlODBC (https://odbc.postgresql.org/)
	ネイティブの接続	はい
Progress OpenEdge	JDBC	JDBC コネクタ (https://www.progress.com/jdbc/openedge)
	ODBC	ODBC コネクタ (https://www.progress.com/odbc/openedge)
SQLite	ネイティブの接続	はい
Sybase	ADO	Sybase ASE OLE DB プロバイダー
	ADO.NET	ASE ADO.NET データプロバイダー
	JDBC	JDBC のための jConnect™
	ODBC	Sybase ASE ODBC ドライバー

** 太字でハイライトされているドライバーは Microsoft により提供されます。Windows システムにより利用することができない場合は、Microsoft Web サイトからダウンロードすることができます。*

ADO.NET プロバイダの一部はサポートされておらず、機能が制限されている場合があります。ADO.NET サポートに関するメモを参照してください。

データベースの種類に関して各データアクセステクノロジーの機能と制限を理解するには、特定のデータベース製品のドキュメントを参照し、特定の環境に対しての接続をテストします。よく発生する接続問題を回避するために、以下の全般的な注意と奨励を考慮してください:

- 32-ビット版と 64 ビット版のドライバーには互換性がない可能性がありますので、Altova アプリケーションに適用することのできる、ドライバーバージョンを構成してインストールしてください。例えば、64 ビット版 オペレーティングシステムで 32 ビット版 Altova アプリケーションを使用している場合、データベース接続は 32 ビット版バージョンのドライバーを使用してください。
- 最新のドライバーのバージョンは、古いバージョンでは使用することのできない機能を含んでいる可能性があります。
- ODBC データソースをセットアップする際、データソース名 (DSN) をユーザー *DSN* の代わりに *システム DSN* として作成することが奨励されます。

- JDBC データソースをセットアップする際、JRE (Java Runtime Environment) がインストールされている、またオペレーティングシステムのCLASSPATH 環境変数が構成されていることを確認してください。
- Microsoft-により提供されるデータベースドライバーに適用することのできるサポートの詳細と既知の問題に関してはMSDN のドキュメントを参照してください。
- データベースベンダーからインストールするドライバーおよびデータベースクライアントソフトウェアのインストールの手順とサポートの詳細は、インストールパッケージのドキュメントを参照してください。公式なまたはサードパーティデータベースドライバーの場合、特定のオペレーティングシステムでの特定のドライバーに適用することのできる最も包括的な情報と構成手順は、通常ドライバーインストールパッケージの一部です。

7.2.1.3 ADO 接続の設定

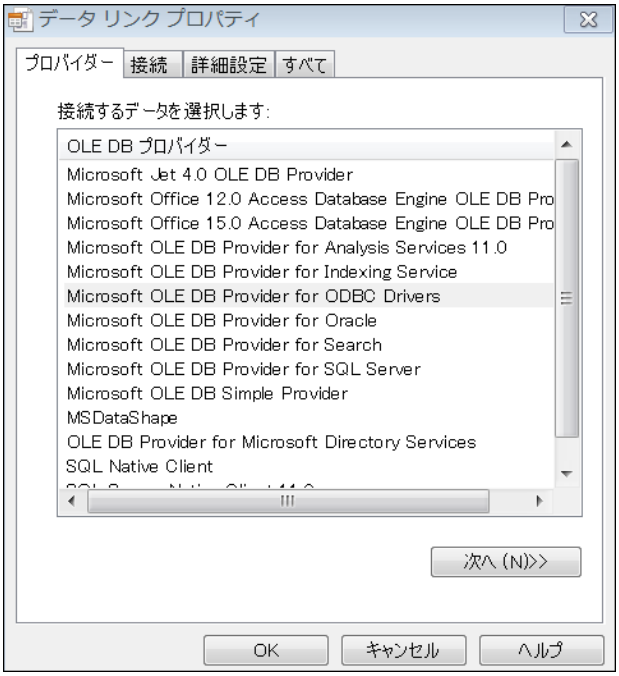
Microsoft ActiveX Data Objects (ADO) は、OLE DB を介して多種のデータソースに接続することのできるデータアクセス技術です。OLE DB は、ODBC またはJDBC の代替インターフェイスので、COM (Component Object Model コンポーネント・オブジェクト・モデル)環境内で、データへの一定のアクセスを提供します。ADO は、Microsoft Access およびSQL Server 等のMicrosoft ネイティブなデータベースへ接続する典型的な選択ですが、他のデータソースにも使用することができます。

ADO 接続のセットアップ:

1. データベース接続ウィザードの開始。
2. **[ADO 接続]** をクリックします。



3. **[構築]** をクリックします。



4. 接続するデータプロバイダーを選択します。下のテーブルは、よくある例をリストしています。

接続するデータベース...	使用するプロバイダー...
Microsoft Access	• **Microsoft Office Access Database Engine OLE DB Provider** Microsoft Access 2003 に接続する際には、**Microsoft Jet OLE DB Provider** も使用することができます。
SQL Server	• **SQL Server Native Client** • **Microsoft OLE DB Provider for SQL Server**
他のデータベース	データベースに適用することのできるプロバイダーを選択します。 データベースへのOLE DB プロバイダーが使用できない場合は、データベースベンダー (データベースドライバー概要を参照) から必要なドライバーをインストールします。別の方法として、ODBC または JDBC 接続をセットアップします。 オペレーティングシステムに必要なデータベースに対するODBC ドライバーがある場合、**Microsoft OLE DB Provider for ODBC Drivers** を使用することができます。

5. **「次へ」** をクリックして、ウィザードを終了します。

次のウィザードのステップは選択されたプロバイダーに固有の手順です。SQL Server には、データベースユーザー名とパスワードに加え、データベースサーバーのホスト名を提供または選択する必要があります。Microsoft Access には、データベースファイルへのパスを参照または提供するように求められます。

初期化プロパティ (接続パラメーター) の完全なリストは、接続ダイアログボックスの**すべて** タブで使用することができます？ これらのプロパティは、選択されたプロバイダーにより異なります。次のセクションは Microsoft Access and SQL Server データベースのための基本的な初期化プロパティの手引きを説明しています：

- SQL サーバー データリンク プロパティのセットアップ
- Microsoft Access データリンク プロパティのセットアップ

既存の Microsoft Access データベースへの接続

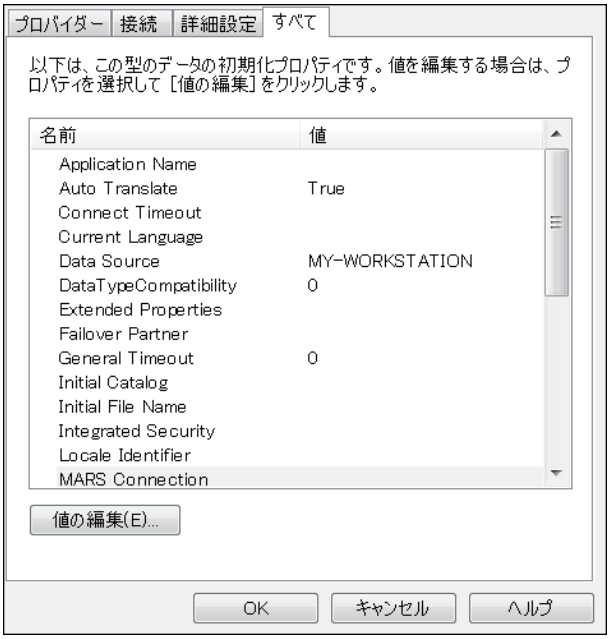
このアプローチは、パスワードで保護されていない Microsoft Access データベースに接続する場合に適しています。データベースのパスワードの設定は、Microsoft Access (ADO) への接続で説明されています。

既存の Microsoft Access データベースへの接続方法：

1. データベース接続ウィザードを実行します（データベース接続ウィザードの開始を参照）。
2. **Microsoft Access (ADO)** を選択して、**「次へ」**をクリックします。
3. データベースファイルを参照、または、（相対または絶対）パスを入力します。
4. **「接続」**をクリックします。

SQL Server データ リンク プロパティの設定

Microsoft SQL Server データベースに ADO (ADO 接続のセットアップを参照) を介して接続する場合、次のデータリンク プロパティが **全て** タブのデータリンク プロパティダイアログボックスで正しく構成されていることを確認してください。

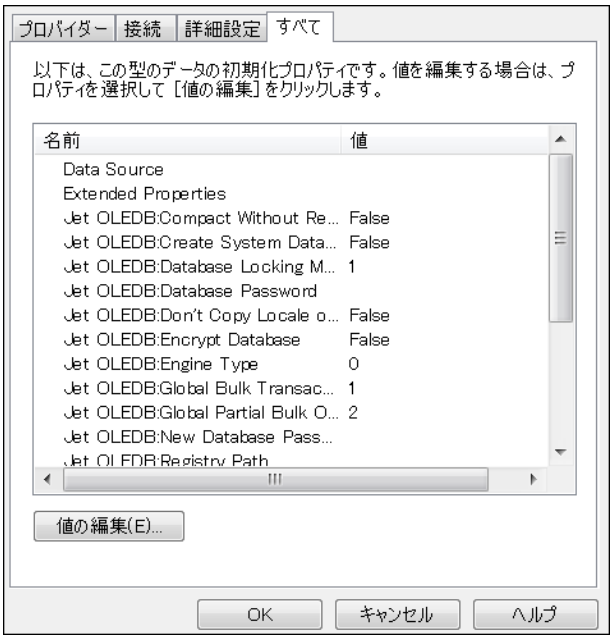


データ リンク プロパティ ダイアログ ボックス

プロパティ	メモ
統合セキュリティ	**プロバイダー** タブで **SQL Server Native Client** がデータプロバイダーに選択された場合は、このプロパティを、空白文字に設定します。
セキュリティ情報を保持	このプロパティを **True** に設定する。

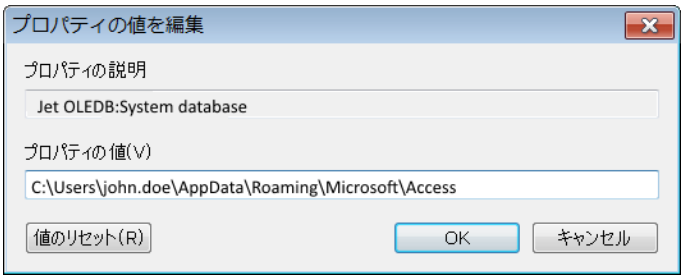
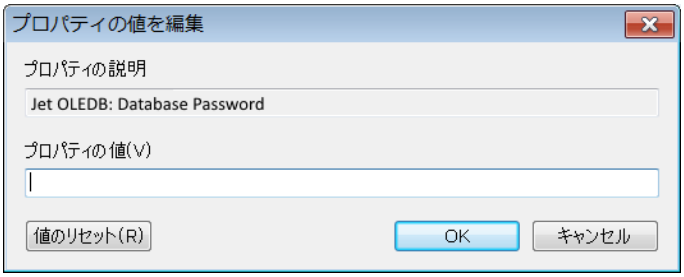
Microsoft Access データ リンク プロパティの設定

Microsoft Access データベースにADO (ADO 接続のセットアップを参照) を介して接続する場合、次のデータ リンク プロパティが**全ての**タブのデータ リンク プロパティ ダイアログ ボックスで正しく構成されていることを確認してください。



データ リンク プロパティ ダイアログ ボックス

プロパティ	メモ
データソース	このプロパティは、Microsoft Access データベース ファイルへのパスを保管します。データベースへの接続問題を回避するために、UNC (Universal Naming Convention 汎用名前付け規則) パス フォーマットの使用が奨励されます。例 : `\\anyserver\share$\filepath`
Jet OLEDB: *システム データベース*	このプロパティはワークグループ情報ファイルへのパスを保管します。データベースは接続する前に、このプロパティを明示的に設定する必要がある可能性があります。 "ワークグループ情報ファイル"エラーが理由で接続できない場合、ワークグループ情報ファイル (**System.MDW**) をユーザー プロファイルに適用できるように設定します (手順はhttp://support.microsoft.com/kb/305542を参照してください)。プロパティの値を**System.MDW** ファイルのパスに設定します。

プロパティ	メモ
	
Jet OLEDB: データベースパスワード	データベースがパスワードで保護されている場合は、このプロパティの値をデータベースのパスワードに設定します。 

7.2.1.4 ADO.NET 接続のセットアップ

ADO.NET は、データベースからのデータを含むデータと作業するためにデザインされたMicrosoft .NET フレームワークライブラリのセットです。MapForce から、データベースに接続するには、ADO.NET、Microsoft .NET フレームワーク 4、または、以降を使用する必要があります。下に表示されているとおり、 NET プロバイダーを選択して、接続文字列を提供し、データベースにADO.NET を介して接続します。

.NET データプロバイダーは、コマンドを実行する特定の型のデータソースへの接続を可能にするクラスのコネクションです（例えば、SQL Server またはOracle データベースなど）。すなわち、ADO.NET とは、MapForce などのアプリケーションがデータベースプロバイダーを介してデータベースと作動することを意味します。各 データプロバイダーは特定の型のデータソースと作動するように最適化されています。NET プロバイダーには以下の2つの種類があります ：

1. デフォルトとしてMicrosoft .NET フレームワークを使用して提供されます。
2. 主要なデータベースベンダーを使用して NET フレームワークの拡張として提供されます。このようなADO.NET プロバイダーは、個別人ストールされている必要があり、通常それぞれのデータベースベンダーのウェブサイトからダウンロードすることができます。

メモ： 特定のADO.NET プロバイダーは、サポートされていない、または、サポートが制約されています。ADO.NET サポートに関するメモを参照してください。

ADO.NET 接続の作成 :

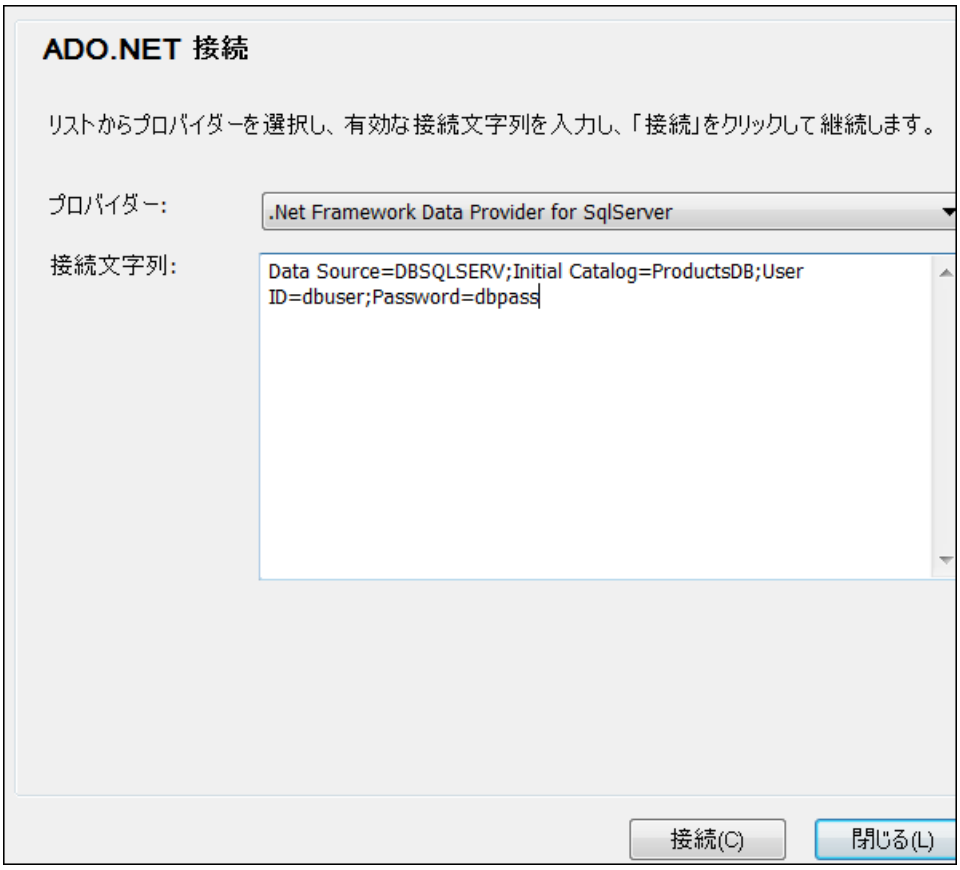
1. データベース接続ウィザードを開始します。
2. 「**ADO.NET 接続**」をクリックします。
3. .NET データ プロバイダーをリストから選択します。

.NET フレームワークと共に使用することのできるプロバイダーのリストはデフォルトでプロバイダーのリストに表示さ

れます。ベンダーに特有の .NET データプロバイダーは、使用中のシステムに既にインストールされている場合のみ使用することができます。 .NET データプロバイダーは、データベースベンダーにより提供される、msi、または、exe ファイルを実行して、GAC (Global Assembly Cache) 内にインストールされる必要があります。

4. データベース接続文字列を入力します。接続文字列は、データベース接続情報をセミコロンで区切られた接続パラメーターのキー値ペアとして定義します。例えば、`Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;password=dbpass` などの文字列は、サーバー `DBSQLSERV` 上で、ユーザー名 `dbuser` とパスワード `dbpass` を使用して、SQL Server データベース `ProductsDB` に接続します。接続文字列を、キー値ペアを直接「接続文字列」ダイアログボックスに入力することにより作成することができます。更に便利な接続オプションは Visual Studio で作成することができます（次を参照：Visual Studio 内で接続文字列を作成する）。

接続文字列の構文は、プロバイダーリストより選択されたプロバイダーにより異なります。例えば、次を参照してください：サンプル ADO.NET 接続文字列。



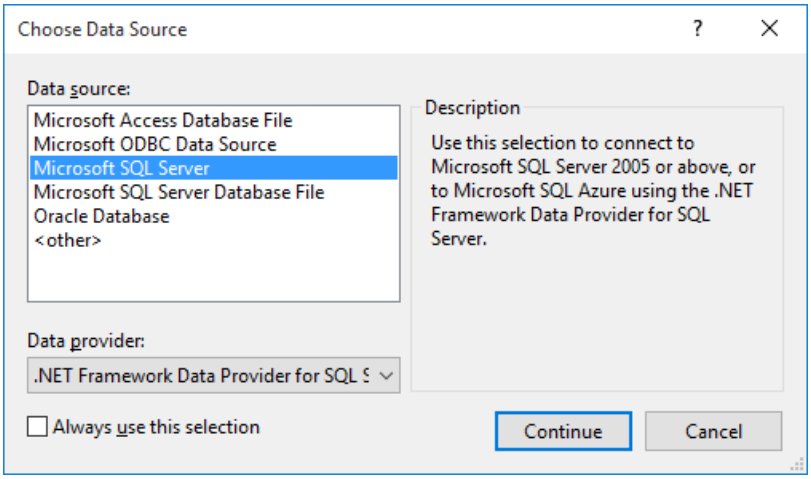
5. **「接続」**をクリックします。

Visual Studio 内で接続文字列を作成する

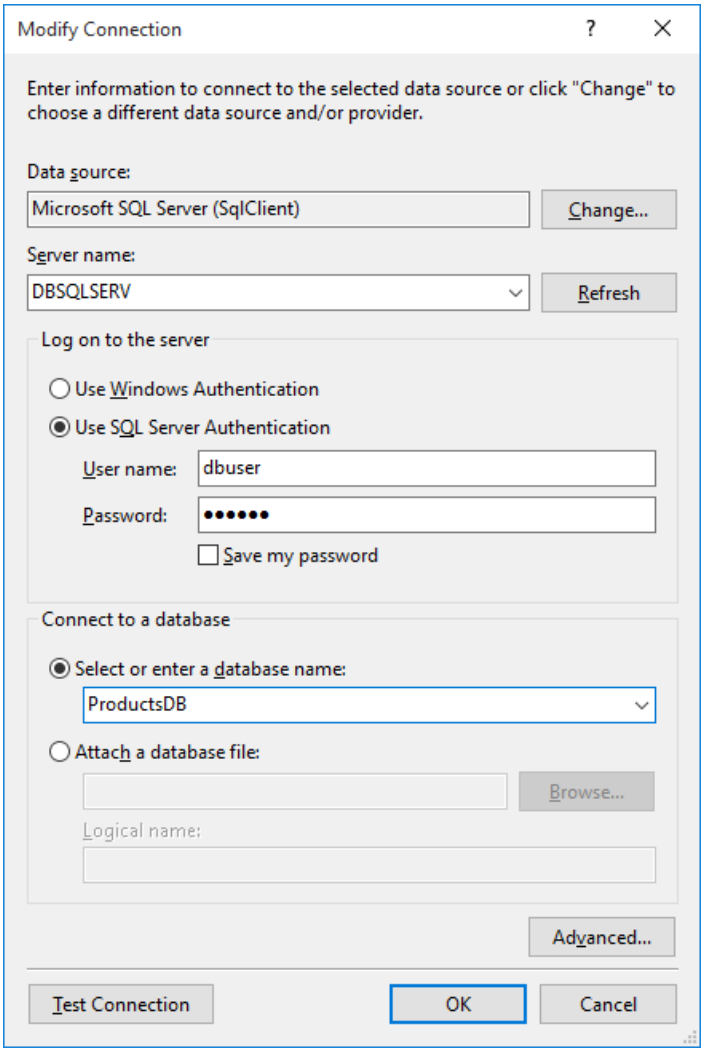
ADO.NET を使用してデータソースに接続するには、有効な接続文字列が必要です。次の手順は、Visual Studio から接続文字列を作成する方法を説明しています。

Visual Studio 内で接続文字列を作成する:

1. 「ツール」メニューから、「データベースへの接続」をクリックします。
2. リストからデータソースを選択します (この例の場合、Microsoft SQL Server)。データプロバイダーは、選択に従い自動的に記入されます。



3. 「接続」をクリックします。



4. データベースへのサーバーホスト名、ユーザー名、およびパスワードを入力します。この例の場合、サーバー DBSQLSERV 上のデータベース ProductsDB に SQL Server 認証を使用して接続します。
5. **「OK」**をクリックします。

データベースへの接続に成功した場合、サーバーエクスプローラー ウィンドウに表示されます。サーバーエクスプローラー ウィンドウを、メニューコマンド **表示 | サーバーエクスプローラー**」を使用して表示することができます。データベース接続文字列を取得するには、サーバーエクスプローラー ウィンドウ内の接続を右クリックし、**「プロパティ」**を選択します。接続文字列が Visual Studio プロパティウィンドウに表示されます。MapForce の 「接続文字列」 ボックスに文字列を貼り付ける前に、アスタリスク (*) 文字を実際のパスワードと置き換える必要があることに注意してください。

サンプル ADO.NET 接続文字列

ADO.NET 接続をセットアップするには、データベース接続ダイアログボックスからADO.NET プロバイダー を選択し、接続文字列を入力します。 (次も参照してください ADO.NET 接続のセットアップ)。異なるデータベースのサンプル ADO.NET 接続文字列は .NET プロバイダーの下にリストされています。

IBM i のための .NET Framework データプロバイダー

このプロバイダーは、*IBM i Access Client Solutions - Windows Application Package* の一部としてインストールされます。サンプル接続文字列は以下のとおりです：

```
DataSource=ServerAddress;UserID=user;Password=password;DataCompression=True;
```

詳細に関しては、インストールパッケージ内に含まれる「.NET プロバイダー技術レファレンス」ヘルプファイルを参照してください。

MySQL のための .NET Framework データプロバイダー

このプロバイダーをMySQL ウェブサイトからダウンロードすることができます（https://dev.mysql.com/downloads/connector/net/）。サンプル接続文字列は以下のとおりです：

```
Server=127.0.0.1;Uid=root;Pwd=12345;Database=test;
```

次も参照してください：https://dev.mysql.com/doc/connector-net/en/connector-net-programming-connecting-connection-string.html

SQL Server のための .NET Framework データプロバイダー

サンプル接続文字列は以下のとおりです：

```
Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User
ID=dbuser;Password=dbpass
```

次も参照してください：https://msdn.microsoft.com/en-us/library/ms254500(v=vs.110).aspx

.NET Framework 4.0 のための IBM DB2 データプロバイダー 10.1.2

```
Database=PRODUCTS;UID=user;Password=password;Server=localhost:50000;
```

メモ： このプロバイダーは、通常 IBM DB2 Data Server Client パッケージと共にインストールされます。IBM DB2 Data Server Client パッケージをインストール後 ADO.NET プロバイダーのリスト内で プロバイダーが不足している場合、次の技術メモを参照してください：http://www-01.ibm.com/support/docview.wss?uid=swg21429586。

次も参照してください：http://www.ibm.com/support/knowledgecenter/en/SSEPGG_10.1.0/com.ibm.swg.im.dbclient.adonet.ref.doc/doc/DB2ConnectionClassConnectionStringProperty.html

.NET (ODP.NET) のための Oracle データプロバイダー

ODP.NET プロバイダーを含むインストールパッケージをOracle ウェブサイトからダウンロードすることができます（次を参照してください：http://www.oracle.com/technetwork/topics/dotnet/downloads/index.html）。サンプル接続文字列は以下のとおりです：

```
Data Source=DSORCL;User Id=user;Password=password;
```

Oracle への接続 (ODBC)内で説明されているように**tnsnames.ora** ファイル内で定義されたOracle サービス名をさす DSORCL がデータソース名の場合。

tnsnames.ora ファイル内のサービス名を構成せず接続する場合は、以下のような文字列を使用してください:

```
Data Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host)
(PORT=port)))(CONNECT_DATA=(SERVER=DEDICATED)
(SERVICE_NAME=MyOracleSID)));User Id=user;Password=password;
```

次も参照してください:https://docs.oracle.com/cd/B28359_01/win.111/b28375/featConnecting.htm

ADO.NET サポートに関するメモ

下のテーブルは、MapForce 内でサポートされていない、または、制限の有るサポートを持つ、既知のADO.NET データベースドライバをリストしたテーブルです。

データベース	ドライバ	サポートに関するメモ
すべてのデータベース	**.Net Framework Data Provider for ODBC**	制限つきサポート。Microsoft Access 接続の際に既知の問題があります。ODBC への直接の接続を代わりに使用することが奨励されます。次を参照してください:ODBC 接続のセットアップ。
	.Net Framework Data Provider for OleDb	制限つきサポート。Microsoft Access 接続の際に既知の問題があります。ADO への直接の接続を代わりに使用することが奨励されます。次を参照してください:ADO 接続のセットアップ。
Firebird	**Firebird ADO.NET Data Provider**	制限つきサポート。ODBC または JDBC を代わりに使用することが奨励されます。次を参照してください:Firebird (ODBC) への接続 と Firebird (JDBC) への接続.
Informix	**IBM Informix Data Provider for .NET Framework 4.0**	サポートされていません。**DB2 Data Server Provider** を代わりに使用してください。
IBM DB2 for i (iSeries)	**.Net Framework Data Provider for i5/OS**	サポートされていません。*IBM i Access Client Solutions - Windows Application Package* の一部として提供されている**.Net Framework Data Provider for IBM i** を代わりに使用してください。
Oracle	**.Net Framework Data Provider for Oracle**	制限つきサポート。.NET Framework と共にこのドライバは提供されていますが、Microsoft により使用を奨励されていません。
PostgreSQL	-	このベンダーのためのADO.NET ドライバはサポートされていません。
Sybase	-	このベンダーのためのADO.NET ドライバはサポートされていません。

7.2.1.5 ODBC 接続の設定

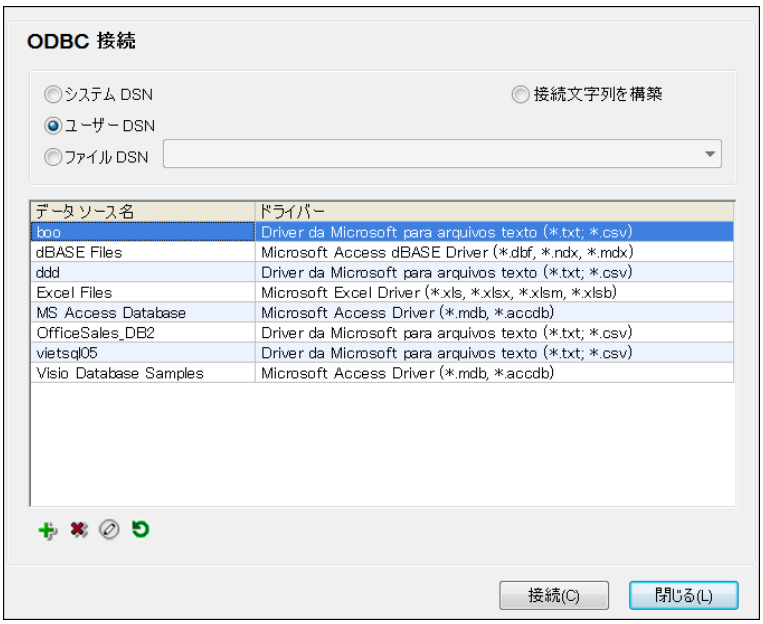
ODBC (Open Database Connectivity) は、一般に使用されているデータアクセス技術で、MapForce からデータベースに接続することができます。データベースにアクセスするための主要な手段として、またはOLE DB- やJDBC-による接続の代替として使用することができます。

ODBC を使用してデータベースに接続するには、最初にODBC データソース名 (DSN) を作成する必要があります。このステップは、他のユーザーまたはオペレーティングシステムによりDSN が既に作成されている場合必要ありません。DSN は、統一された、MapForce を含むオペレーティングシステム上のODBC に対応したクライアントアプリのデータベース接続方法を表します。DSN には以下の種類があります :

- システムDSN
- ユーザーDSN
- ファイルDSN

オペレーティングシステムで特権を持つすべてのユーザーはシステム データソースにアクセスすることができます。作成したユーザーのみがユーザー データソースにアクセスすることができます。 ファイルDSN を作成した場合、データソースは dsn 拡張子を持つファイルとして作成され、データソースにより使用されているドライバーがインストールされていることが前提で、他のユーザーと共有することができます。

ODBC 接続ダイアログボックス内の**[ODBC 接続]** をクリックすると、使用中のコンピューターで既に使用することのできるDSN はデータベース接続ダイアログボックスに表示されています。



ODBC 接続ダイアログボックス

データベースにより必要とされるDSN が無い場合、MapForce データベース接続ウィザードが作成の手助けをします。しかし、Windows オペレーティングシステムに直接作成することもできます。いずれにせよ、続行する前に、使用中のデータベースにODBC ドライバーがオペレーティングシステムのODBC ドライバーリストに適用できることを確認してください。 (使用可能なODBC ドライバーの表示を参照)。

新規 DSN を使用しての接続 :

1. データベース接続ウィザードを開始します。

2. データベース接続ダイアログボックスの **[ODBC 接続]** をクリックします。
3. ソースの種類を選択します (ユーザー DSN、システム DSN、ファイル DSN)。

> システム DSN を作成するには、オペレーティングシステムの管理者権限が必要です。

4. **「追加」** をクリックします。
5. ドライバーを選択して、(作成する DSN の種類に応じて) **[ユーザー DSN]** または **[システム DSN]** をクリックします。使用中のデータベースに適用することのできるドライバーがリストされていない場合、データベースベンダーからダウンロードして、インストールしてください。(次を参照 : データベースドライバー概要)。
6. ポップアップするダイアログボックスに特定の接続情報を入力してセットアップを完了します。

接続を成功させるには、データベースサーバーのホスト名 (または IP アドレス)、とデータベースのユーザー名およびパスワードを提供する必要があります。他の接続パラメーターがある可能性がありますが、これらのパラメーターはデータベースプロバイダーにより異なります。各接続方法に固有のパラメーターについての詳細は、ドライバープロバイダーのドキュメントを参照してください。一度作成されると、DSN はデータソース名のリストで使用することができます。ユーザー DSN はユーザー DSN のリストに追加されますが、システム DSN はシステム DSN のリストに追加されます。

既存の DSN を使用しての接続 :

1. データベース接続ウィザードを開始します.
2. **[ODBC 接続]** をクリックします。
3. 既存のデータソース (ユーザー DSN、システム DSN、ファイル DSN) から種類を選択します
4. 既存の DSN をクリックして、 **「接続」** をクリックします。

既存の dsn ファイルをベースにして接続文字列を構築する :

1. データベース接続ウィザードを開始します.
2. **[ODBC 接続]** をクリックします。
3. **[接続文字列の構築]** を選択して、**[構築]** をクリックします。
4. ファイル DSN を使用して、接続文字列を構築する場合、**[ファイルデータソース]** タブをクリックします。それ以外の場合は、**[マシンデータソース]** タブをクリックします。(システム DSN とユーザー DSN は "マシン" データソースとして知られています。)
5. 必要とされる dsn ファイルを選択して、 **「OK」** をクリックします。

用意された接続文字列を使用して接続する :

1. データベース接続ウィザードを開始します.
2. **[ODBC 接続]** をクリックします。
3. **[接続文字列の構築]** を選択しています。
4. 接続文字列を与えられたボックスに貼り付け、 **「接続」** をクリックします。

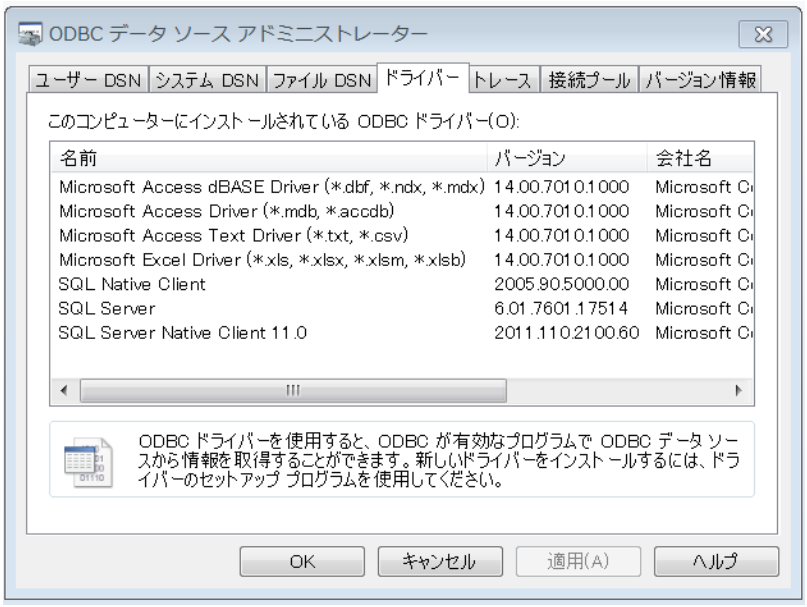
使用可能な ODBC ドライバーの表示

ODBC データソース管理で使用中のオペレーティングシステムで使用可能な ODBC ドライバーを確認することができます。Windows コントロールパネルの **管理ツール** から ODBC データソース管理 (**Odbcad32.exe**) にアクセスすることができます。64 ビット版オペレーティングシステムには、2 種類のバージョンの実行可能ファイルがあります :

- **C:\Windows\SysWoW64** ディレクトリにある、**Odbcad32.exe** ファイルの 32 ビット版バージョン (**C:** が使用中のシステムドライブとして前提されています)。
- **C:\Windows\System32** ディレクトリにある、**Odbcad32.exe** ファイルの 64 ビット版バージョン

インストールされている 32 ビット版データベースドライバーは、ODBC データソース管理の 32 ビット版バージョンで、64-ビット版バージョン内の 64 ビット版ドライバーで確認することができます。ですから、対応するバージョンの ODBC デー

タソース管理でデータベースドライバーを確認してください。



ODBC データソース管理者

ターゲットデータベースのドライバーがリストに存在しない場合、または、代替ドライバーを追加する場合、データベースのベンダーからダウンロードしてください (データベースドライバー概要 を参照)。ODBC ドライバーがシステムで使用可能になると、ODBC 接続を作成することができます (ODBC 接続のセットアップを参照)。

7.2.1.6 JDBC 接続の設定

JDBC (Java Database Connectivity) は、Oracle の Java ソフトウェアプラットフォームの一部であるデータベースアクセスインターフェイスです。JDBC 接続は ODBC 接続に比べ、通常よりもリソースを集中的に使用する接続ですが、ODBC で使用することのできない機能を使用することができます。ODBC コネクタにより使用できないデータベース機能を使用する場合 JDBC 接続を使用することが奨励されています。

前提条件

- Java Runtime Environment (JRE) または Java Development Kit (JDK) がオペレーティングシステムにインストールされている必要があります。インストールされていない場合は Java の公式 Web サイトからパッケージとインストールの手順をダウンロードしてください。
- データベースベンダーからの JDBC ドライバーがインストールされている必要があります。Oracle データベースに接続する場合は、一部の Oracle ドライバーは特定の JRE バージョンに固有のため、追加コンポーネントと構成が必要な点に注意してください。Oracle 製品のドキュメント (例えば、『JDBC 開発者ガイドおよびリファレンス』) には、各 JDBC ドライバーの構成の手続きの詳細が説明されています。
- オペレーティングシステムの `PATH` 環境変数は、JRE または JDK インストールディレクトリの `bin` ディレクトリへのパスを含む必要があります。例えば、`C:\Program Files (x86)\Java\jre1.8.0_51\bin` など。
- `CLASSPATH` 環境変数は、(jar ファイルの 1 つまたは複数の) 使用中の Windows オペレーティングシステムの JDBC ドライバーのパスを含む必要があります。データベースクライアントをインストールする際に、インストーラーはこの変数を自動的に構成する可能性があります。JDBC ドライバーのドキュメントは通常ステップバイステップで CLASSPATH 変数の設定方法を説明しています (CLASSPATH の構成を参照)。

JDBC 接続の設定方法

1. データベース接続ウィザードを開始します。

2. 『**DBC 接続**』をクリックします。
3. 任意で、セミコロンで区切られた jar ファイルパスのリストを『クラスパス』テキストボックスに入力することもできます。ここで入力される.jar ライブラリは、CLASSPATH 環境変数内で既に定義されているものに加え、環境にロードされます。『クラスパス』テキストボックスの編集が完了すると、ソース jar ライブラリ内で検索されるJDBC ドライバーは、『ドライバー』リストに自動的に追加されます（次のステップを参照してください）。
4. 『ドライバー』の横でJDBC ドライバーをリストから選択、または、Java クラス名を入力します。CLASSPATH 環境変数により構成されたJDBC ドライバーおよび、『クラスパス』テキストボックス内で検出されたドライバーが含まれます（次を参照：CLASSPATH の構成）。

> CLASSPATH 変数内で定義されているJDBC ドライバーのパスを使用することができます。また、データベース接続ダイアログボックスに直接入力された jar パスはJava Virtual Machine (JVM) によりサポートされています。JVM は接続を構築するためのドライバーを決定します。JVM にロードされるJava クラスを確認し、JDBC ドライバーの競合、および、データベースに接続する際に発生する予期されないエラーを回避するようにしてください。

5. 対応するボックスに、ユーザー名とパスワードを入力します。
6. データベースURL テキストボックス内に、JDBC 接続 URL (文字列) をデータベースの型のとおり入力します。次のテーブルは一般のデータベースの型のためのJDBC 接続 URL (文字列) の構文について説明しています。

データベース	JDBC 接続 URL
Firebird	`jdbc:firebirdsql://<host>[:<port>]/<database path or alias>`
IBM DB2	`jdbc:db2://hostName:port/databaseName`
IBM Informix	`jdbc:informix-sqli://hostName:port/databaseName:INFORMIXSERVER=myserver`
Microsoft SQL Server	`jdbc:sqlserver://hostName:port;databaseName=name`
MySQL	`jdbc:mysql://hostName:port/databaseName`
Oracle	`jdbc:oracle:thin:@//hostName:port:service`
Oracle XML DB	`jdbc:oracle:oci:@//hostName:port:service`
PostgreSQL	`jdbc:postgresql://hostName:port/databaseName`
Progress OpenEdge	`jdbc:datadirect:openedge://host:port;databaseName=db_name`
Sybase	`jdbc:sybase:Tds:hostName:port/databaseName`

メモ： 上記のフォーマットに対する構文のバリエーションも使用することができます（例えば、データベースURL はポートを除外、またはデータベースのユーザー名とパスワードを含む可能性があります）。詳細はデータベースのベンダーのドキュメントを参照してください。

7. **接続**』をクリックします。

CLASSPATH の構成

CLASSPATH 環境変数はクラスと、オペレーティングシステム上の他のリソースファイルを検索するためJava Runtime Environment (JRE) により使用されます。JDBC を介してデータベースに接続すると、この変数はオペレーティングシステ

ム上のJDBC ドライバーへのパス、または一部の場合、使用中のデータベースの種類に固有の追加ライブラリファイルへのパスを含むように構成されなければなりません。

下のテーブルは、CLASSPATH 変数に通常含まれる必要のあるサンプルファイルパスをリストしています。重要なことは、システム上のJDBC ドライバのロケーション、JDBC ドライバー名、オペレーティングシステム上のドライバーの場所をベースにこの情報を調整する必要があります。接続問題を避けるために、使用中のオペレーティングシステムにインストールされているJDBC ドライバに適用されるインストールの指示、インストール前とインストール後の構成ステップを確認してください。

データベース	**サンプルCLASSPATH エントリ**
IBM DB2	`C:\Program Files (x86)\IBM\SQLLIB\java\db2jcc.jar;C:\Program Files (x86)\IBM\SQLLIB\java\db2jcc_license_cu.jar;`
IBM Informix	`C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;`
Microsoft SQL Server	`C:\Program Files\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar`
MySQL	`mysql-connector-java-`*`version`*`-bin.jar;`
Oracle	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`
Oracle (with XML DB)	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`**`ORACLE_HOME`**`\LIB\xmlparserv2.jar;`**`ORACLE_HOME`**`\RDBMS\jlib\xdb.jar;`
PostgreSQL	**`<installation directory>`**`\postgresql.jar`
Sybase	`C:\sybase\jConnect-5_5\classes\jconn2.jar`

- CLASSPATH 変数の変更は、使用中のコンピュータのJava アプリケーションの振る舞いに影響する可能性があります。続行する前に生じえる影響を把握するために、Java ドキュメントを参照してください。
- 環境変数は、ユーザーまたはシステムであることができます。システム環境変数を変更するには、オペレーティングシステムに対する管理者権限が必要です。
- 環境変数を変更後、設定の変更を有効にするには、プログラムを再起動してください。または、オペレーティングシステムをログオフまたは再起動してください。

Windows 7 でCLASSPATH を構成する:

1. **[スタート]** メニューを開始して、**[コンピューター]** を右クリックします。
2. **[プロパティ]** をクリックします。
3. **[システムの詳細設定]** をクリックします。
4. **[詳細設定]** タブから**[環境変数]** をクリックします。
5. CLASSPATH 変数を、ユーザーまたはシステム環境変数からロケートして、編集をクリックします。
   CLASSPATH 変数が存在しない場合は、**[新規]** をクリックして作成します。
6. オペレーティングシステム内でJDBC ドライバーがある場所にパスを含むように変数の値を編集します。JDBC ドライバーパスを他の既にCLASSPATH 変数内にあるパスから区別するためにセミコロン(;)で区切ります。

Windows 8、Windows 10 でCLASSPATH を構成する:

1. Windows のスタートボタンを右クリックして、**[システム]** をクリックします。
2. **[高度なシステム設定]** をクリックします。
3. **[環境変数]** をクリックします。
4. CLASSPATH 変数を、ユーザーまたはシステム環境変数からロケートして、編集をクリックします。
   CLASSPATH 変数が存在しない場合は、**[新規]** をクリックして作成します。

5. オペレーティングシステム内でJDBC ドライバーがある場所にパスを含むように変数の値を編集します。JDBC ドライバーパスを他の既にCLASSPATH 変数内にあるパスから区別するためにセミコロン(;)で区切ります。

7.2.1.7 PostgreSQL 接続のセットアップ

PostgreSQL データベースへの接続は、ネイティブな接続として、または、ODBC、JDBC、および他のドライバーを使用して確立することができます。ネイティブな接続の利点は、使用中のシステムにインストールする場合、ドライバーを必要としないことです。また、Linux、または、OS X サーバー上での実行のためにファイルをデプロイする場合、ターゲットサーバー上にドライバーをインストールする必要はありません (次も参照してください:Linux とMac 上でのデータベース接続)。

ドライバーを使用して接続を構築する場合は、次を参照してください:データベースドライバーの概要. JDBC を使用しての接続に関しては、次を参照してください:JDBC 接続のセットアップ。ODBC を使用したPostgreSQL への接続の例は、次を参照してください:PostgreSQL (ODBC) への接続 。

それ以外の場合、PostgreSQL へのネイティブの接続をセットアップする場合、以下の手順に従ってください。継続するには、以下の前提条件が必要です:ホスト名、ポート、データベース名、ユーザー名、およびパスワード。

ネイティブPostgreSQL 接続のセットアップ:

1. データベース接続ウィザードを開始します.
2. **「PostgreSQL 接続」**をクリックします。
3. 対応するボックスに、ホスト(PostgreSQL が同じマシンで作動している場合、*ローカルホスト*)、ポート(通常、5432 ですが、これは任意です)、データベース名、ユーザー名とパスワードを入力します。



4. **「接続」**をクリックします。

PostgreSQL データベースサーバーが異なるマシンに存在する場合、次の点に注意してください:

- PostgreSQL データベースサーバーは、クライアントからの接続を受け入れるように構成されなければなりません。具体的には、**pg_hba.conf** ファイルがローカルではない接続を許可するように構成される必要があります。第2に、**postgresql.conf** ファイルは、特定されたIP アドレスとポートをリッスンするように構成される必要があります。詳細に関しては、PostgreSQL ドキュメントを参照してください (https://www.postgresql.org/docs/9.5/static/client-authentication-problems.html)。

- サーバーマシンがファイアウォールを越えてデザインされたポート (通常、5432) 上で接続を受け入れるように構成されている必要があります。例えば、**「コントロールパネル > Windows ファイアウォール > 詳細設定 > 受信の規則」**で設定すると、Windows 上で作動するデータベースサーバーでは、ファイアーウォールを越えてポート 5432 に接続を許可するルールを作成する必要がない場合があります。

7.2.1.8 SQLite 接続の設定

SQLite (http://www.sqlite.org) は、ポータブル性とインストールの簡便性が重要な場合理想的な構成のファイルベースの、内蔵のデータ型です。Since SQLite データベースは、MapForce によりネイティブにサポートされるため、接続のためにドライバーをインストールする必要はありません。また、以下の点に注意してください:

- (マッピングをプレビューまたは MapForce Server 実行ファイルを実行する際) SQLite データベースは MapForce 内蔵 変換言語によりサポートされています。
- ユーザー定義関数 (UDF) は SQLite データベースをサポートしません。
- Linux では、SQLite データベースのステートメント実行タイムアウトはサポートされません。
- フルテキスト検索テーブルはサポートされません。
- テーブルの各行内の異なるデータ型の値を SQLite は許可します。MapForce では、処理されたすべての値は宣言された列の型との互換性が必要です。ですから、SQLite データベースが宣言された列の値と異なる行の値を持つ場合ランタイムエラーが起る可能性があります。
- SQLite データベースにマッピングを書き込む場合、また、ターゲットデータベースが存在しない場合、外部にターゲットデータベースを作成する必要があります。この場合、DatabaseSpy (http://www.altova.com/ja/databasespy.html) などのツールを使用して作成するか、または SQLite コマンドラインシェルを公式の Web サイトからダウンロードして、コマンドラインからデータベースファイルを作成してください (サンプル: XML から SQLite へデータをマップを参照)。SQLite SQLite コマンド構文に関する完全なレファレンスは、SQLite の公式のドキュメントを参照してください。

既存の SQLite データベースへの接続

既存の SQLite データベースへの接続方法:

1. データベース接続ウィザードを実行します (データベース接続ウィザードの開始を参照)。
2. **SQLite** を選択して、**「次へ」**をクリックします。
3. SQLite データベースファイルを参照して、または、(相対または絶対) パスを入力します。**「接続」**ボタンは、データベースファイルへのパスを入力すると有効化されます。
4. **「接続」**をクリックします。

サンプル XML から SQLite へのデータマッピング

このサンプルは、XML ファイルからデータを読み込み SQLite データベースに書き込む MapForce マッピングを作成するために必要なステップを説明します。このサンプルには、サンプルマッピングデザイン (mfd) ファイルがあります。このサンプルを開始する前に、サンプルファイルを確認するには、以下のパスからファイルを開いてください: **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\XMLtoSQLite.mfd**.

このサンプルの目的は、XML ファイルからデータを SQLite データベースに挿入することです。サンプルの目的を達成するために、データの書き込みができる空の SQLite データベースが必要です。このサンプルのスコープのために、公式の SQLite Web サイトから使用することのできるコマンドラインシェルを利用して SQLite データベースを作成します。また、この手順は他のツールを使用して行うことができます。

SQLite データベースの作成方法：

1. SQLite ダウンロードページ (http://www.sqlite.org/download.html) から、Windows のための SQLite コマンドラインシェルをダウンロードし、.zip アーカイブをローカルコンピューターでアンパックします。ます。(このサンプルのスコープには **c:\sqlite** を使用します)。
2. **c:\sqlite\sqlite3.exe** を実行して、以下のステートメントを入力します：

```
create table articles (number smallint, name varchar(10), singleprice
real);
```

これによりインメモリデータベース内に `articles` が作成されます。テーブル `articles` は3つの列により構成されます：*number*、*name*、および *singleprice*。これらの列の目的は、ソース XML スキーマ内で定義された同じ名前を持つ要素からのデータを保管することです。各列は列内で保管されるデータに適切なデータ型と共に宣言されています。

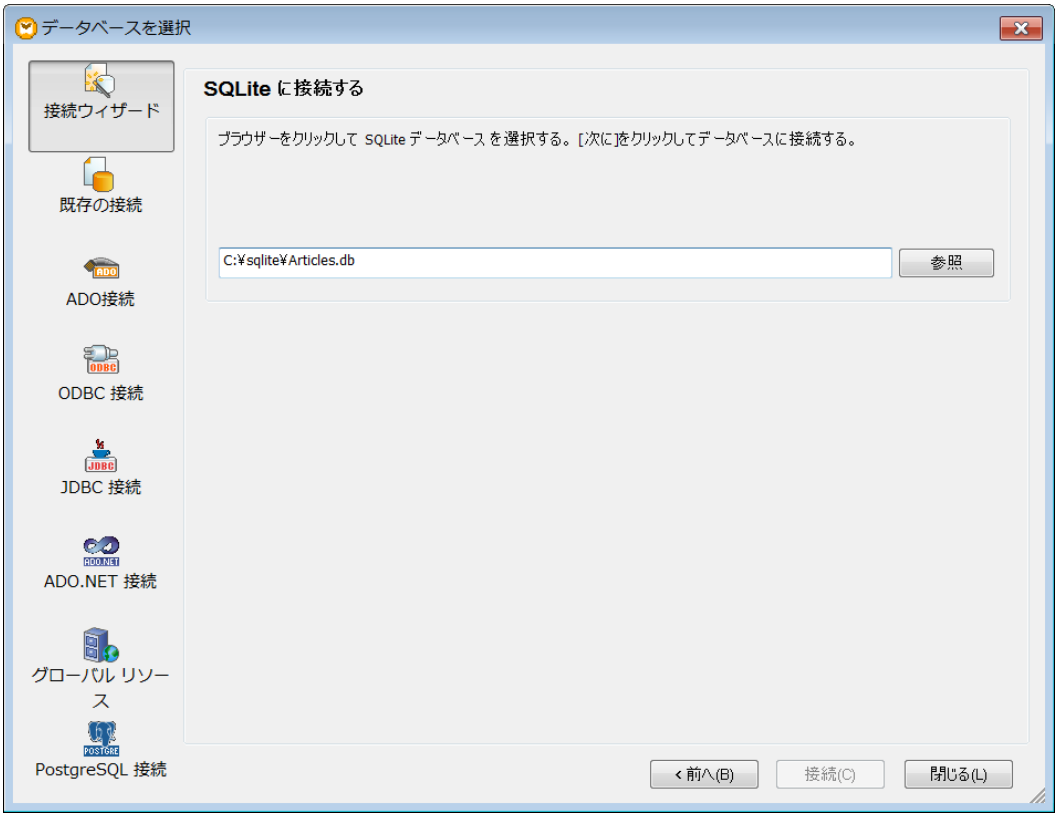
3. コマンドを実行します：

```
.save Articles.db
```

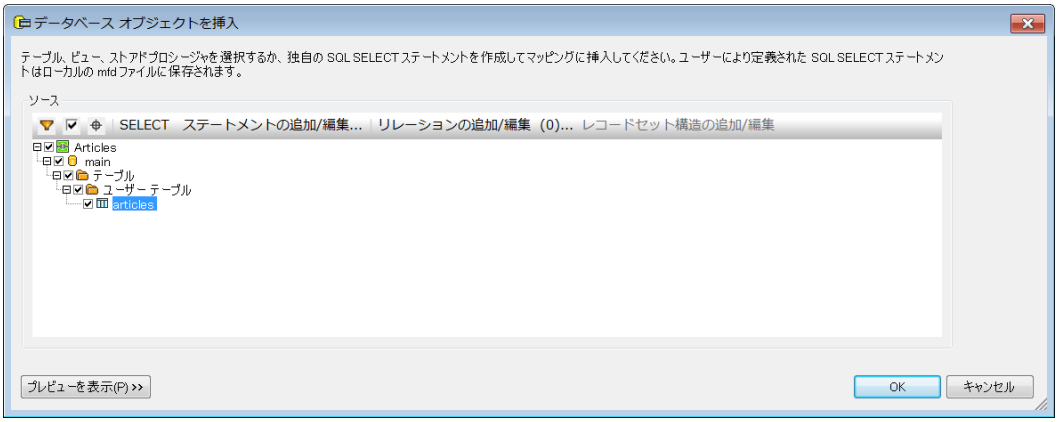
これにより、インメモリデータベースを現在の作業パス：**c:\sqlite\Articles.db** に保存します。次のステップでこのパスを参照することにご注意してください。

XML から SQLite マッピングデザインの作成方法：

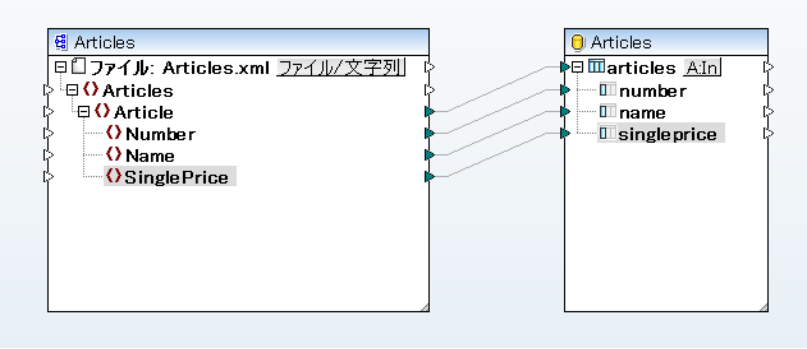
1. MapForce を実行し、変換言語が内蔵されているに設定されていることを確認してください (メニューコマンド **[出力 | 内蔵された実行エンジン]** を使用します)。
2. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にあるファイル **Articles.xml** にマッピングエリアを追加します。(メニューコマンド **[ファイル | XML スキーマ/ファイルの挿入]** を使用します)。
3. 前のステップで作成されたデータベース **Articles.db** のマップエリアに追加し (メニューコマンド **[ファイル | データベースの挿入]** を使用します) **SQLite** を選択します。



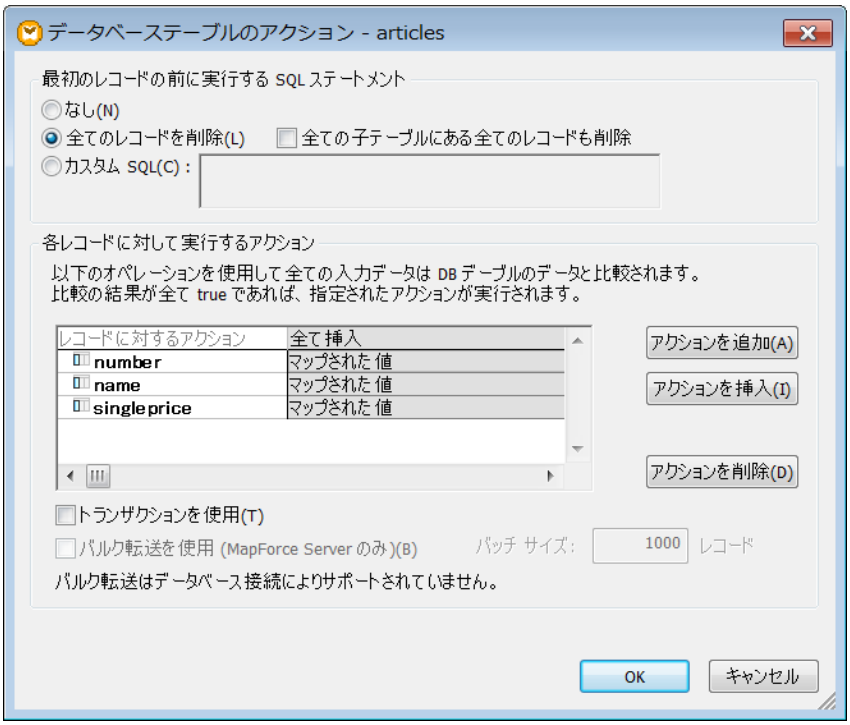
4. データベースオブジェクトを選択するようにプロンプトされると、`articles` テーブルを選択します。



5. 以下に表示されるように接続を作成します：



6. データベースコンポーネントの **A:In** ボタンをクリックし、**[すべての記録の削除]** オプションを選択します。マッピングが実行されると、重複を避けるために、すべての既存のデータベースの行は削除されます。



7. メインマッピングウィンドウの **[出力]** タブをクリックします。MapForce は、マッピングを内蔵された実行エンジンを使用して行い、SQL クエリを出力ウィンドウで表示します。

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   C:\sqlite\Articles.db
6   */
7
8   PRAGMA foreign_keys = ON;
9
10  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)
11
12  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)
13
14  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)
15
16  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
17
```

8. (メニューコマンド**[出力 | SQL スクリプトの実行]** を使用して)SQL スクリプトを実行して、データベースを作成します。MapForce がランタイムエラーに遭遇しない場合、記録はSQLiteデータベースに挿入されます。

SQLite データベースに正しくデータが挿入されたかを確認する:

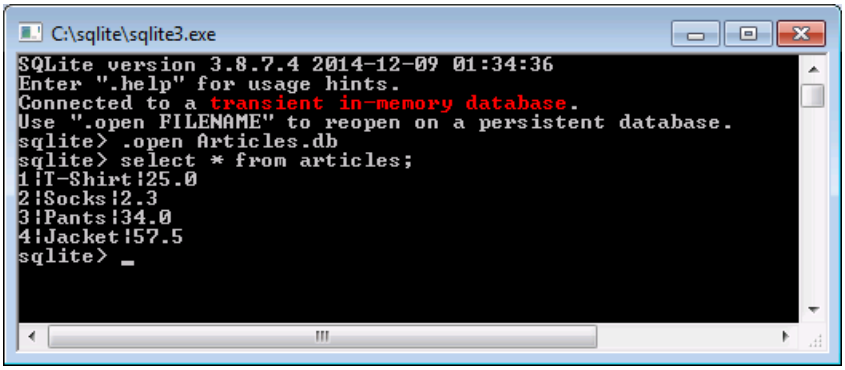
1. **c:\sqlite\sqlite3.exe** を実行して、コマンドを使用してデータベースを開きます:

```
.open Articles.db
```

2. select ステートメントを実行します:

```
select * from articles;
```

このクエリは (¦) 文字により区切られた`articles` テーブル内のすべての記録を返します:



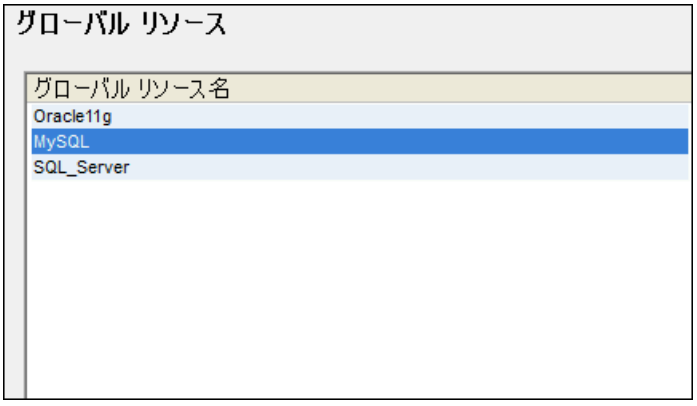
上記のとおり、クエリは 4 つの行を返します。これは、このサンプルの意図した目的であるソースファイルの記録と一致します。

7.2.1.9 グローバル リソースからの接続の使用

以前にデータベースをグローバルリソースとして使用可能に構成している場合、接続をいつでも再使用することができます (異なるAltova アプリケーション間でも使用可能)。

グローバルリソースからのデータベース接続の使用:

1. データベース接続ウィザードを開始します。
2. **[グローバルリソース]** をクリックします。グローバルリソースとして使用可能なデータベース接続がリストされます。



3. データベース接続履歴から **[接続]** をクリックします。

ヒント: 各グローバルリソースの追加情報を取得するために、マウスカーソルをグローバルリソースにポイントします。

7.2.1.10 データベース接続のサンプル

このセクションは、MapForce からデータベースへの接続のサンプル手順を説明します。Windows マシン、ネットワーク環境、とデータベースクライアントまたはサーバーソフトウェアは、次に挙げられるサンプルの構成とは完全に一致しない可能性があります。

メモ: ほぼすべてのデータベースの型では、1つ以上のデータアクセステクノロジー (ADO, ODBC, JDBC) またはドライバーを使用して接続することができます。データベース接続パフォーマンス、およびドライバーの機能と制限は、MapForce 外で構成された選択されたドライバー、(該当する場合) データベースクライアントソフトウェア、および追加接続パラメータなどによって異なります。

Firebird (ODBC) への接続

このトピックは、Linux サーバー上で実行される、Firebird 2.5.4 データベースへの接続のサンプルの命令を説明しています。

必要条件:

- Firebird データベースサーバーは、クライアントからの TCP/IP 接続を受け入れるように構成されています。
- Firebird ODBC ドライバーを使用中のオペレーティングシステムにインストールする必要があります。このサンプルは、を使用します。 Firebird ODBC ドライバー バージョン 2.0.3.154 は、Firebird ウェブサイト (http://www.firebirdsql.org/) からダウンロードすることができます。
- Firebird クライアントを、使用中のオペレーティングシステムにインストールする必要があります。 Firebird 2.5.4 クライアントのためのスタンドアロンのインストーラーが存在しない点に注意してください。また、クライアントは、サーバーのインストールパックの一部です。Firebird サーバー インストールパックを Firebird ウェブサイト (http://www.firebirdsql.org/) からダウンロードすることができます。ダウンロードの際、[スーパークラシック/クラシックまたはスーパーサーバーのための、Windows 実行ファイル] インストーラーを選択してください。クライアントファイルのみをインストールする場合、**[最小クライアントインストール - サーバーツール無し]** を選択します。

  重要点:

  - Firebird ODBC ドライバーおよびクライアント (32 ビットまたは 64 ビット) のプラットフォームは、MapForce のプラットフォームに適応する必要があります。
  - Firebird クライアントのバージョンは、接続されている Firebird サーバーのバージョンに適応する必要があります。

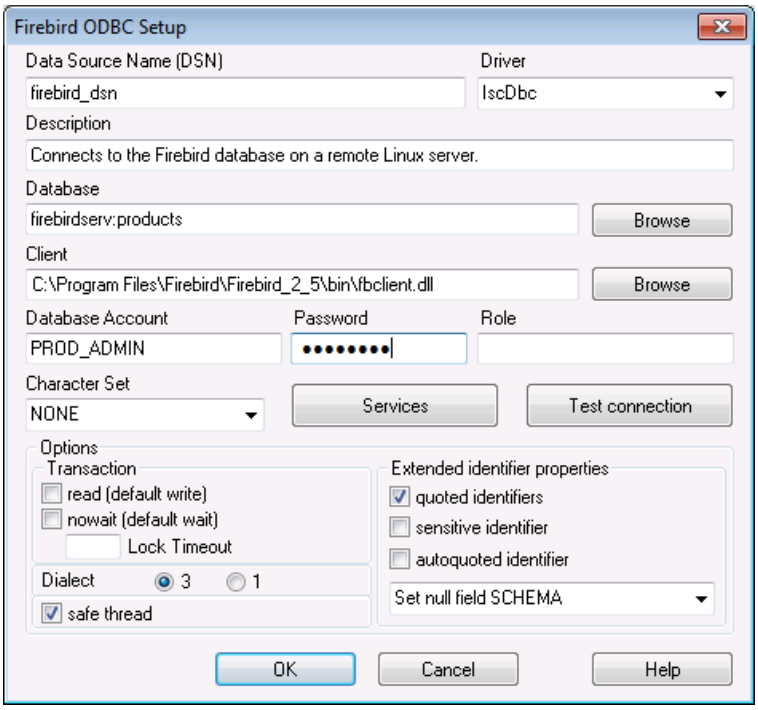
- データベース接続の詳細 サーバー上の サーバー ホスト名 または IP アドレス、データベース パス (または エイリアス)、ユーザー名、および パスワード。

ODBC を使用して Firebird に接続する:

1. データベース接続ウィザードを開始します。
2. **[ODBC 接続]** をクリックする。
3. **[ユーザー DSN]** を選択して、(または、管理者の特権がある場合、 **[システム DSN]** を選択して、**[追加]** をクリックします。



4. Firebird ドライバーを選択して、をクリックします。そして**[ユーザー DSN]** (前のステップでの選択により、または**[システム DSN]**)。Firebird ドライバーがリスト内に存在しない場合、使用中のオペレーティングシステムにインストールされていることを確認してください (使用可能な ODBC ドライバーの表示 を参照)。



5. データベース接続の詳細を入力します:

データソース名 (DSN)	作成する詳細名とデータソースを入力します。
データベース	サーバー ホスト名 または IP アドレスをコロンとデータベース エイリアス (またはパス) の後に入力します。このサンプルでは、ホスト名は、`firebirdserv`、データベースエイリアスは、`products` です: `firebirdserv:products` データベース エイリアスを使用することにより、サーバー側上では、データベース管理者は、エイリアス製品を構成し、サーバー上の実際の Firebird (.fdb) データベースファイルをポイントします (詳細は、Firebird のドキュメントを参照してください)。

	ホスト名の代わりに サーバー IP アドレス、エイリアスの代わりにパスを使用することもできます。ですから、以下のサンプル接続の文字列は有効です：
	
	`firebirdserver:/var/Firebird/databases/butterflies.fdb`
	`127.0.0.1:D:\Misc\Lenders.fdb`
	データベースが Windows マシン上に存在する場合、**[参照]** をクリックして、Firebird (.fdb) データベースファイルを直接選択します。

クライアント	**fbclient.dll** ファイルへのパスを入力します。デフォルトでは、これは Firebird インストールディレクトリのサブディレクトリです。
データベース アカウント	データベース管理者 (このサンプルの場合 `PROD_ADMIN`) により与えられたデータベースユーザー名を入力します。
パスワード	データベース管理者により与えられたデータベースパスワードを入力します。

6. **「OK」** をクリックします。

Firebird (JDBC) への接続

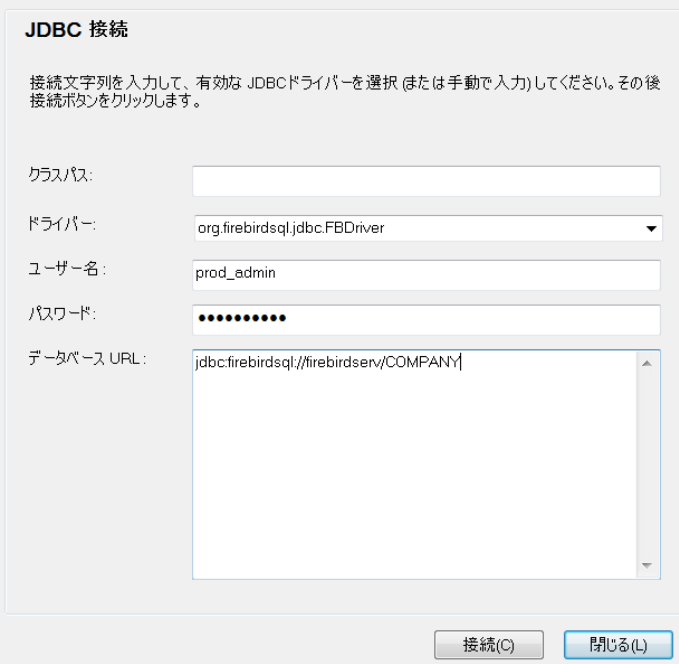
このトピックは、を使用してのFirebird データベースサーバーへの接続のサンプルの命令を説明しています。

必要条件：

- Java Runtime Environment (JRE) またはJava Development Kit (JDK) が使用中のオペレーティングシステムにインストールされている必要があります。
- オペレーティングシステムの `PATH` 環境変数は、JRE またはJDK のインストールディレクトリの `bin` ディレクトリへのパスを含む必要があります。例えば、`C:\Program Files (x86)\Java\jre1.8.0_51\bin`.
- Firebird JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります (データベースへの接続を可能にする jar ファイルのフォームをとります)。ドライバーはFirebird Web サイトからダウンロードすることができます (http://www.firebirdsql.org/)。このサンプルは、*Jaybird 2.2.8* を使用します。
- データベース接続の詳細があります：ホスト、データベースパス、またはエイリアス、ユーザー名、およびパスワード。

JDBC を使用して Firebird に接続する：

1. データベース接続ウィザードを開始します.
2. **「JDBC 接続」** をクリックします。.
3. 「クラスパス」の横にデータベースへの接続を与える jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる jar ファイルは次のパスを使用して見つけることができます：**C:\jdbc\firebird\jaybird-full-2.2.8.jar**。jar ファイルパスをオペレーティングシステムのCLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照：CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**org.firebirdsql.jdbc.FBDriver** を選択します。このエントリは、有効な jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムのCLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。



5. 対応するテキストボックスに、データベースへのユーザー名とパスワードを入力します。
6. URL テキストボックスに、データベースサーバーに適用することのできるハイライトされた値を置き換える、データベースサーバーの接続文字列を入力します。

```
jdbc:firebirdsql://<host>[:<port>]/<databese path or alias>
```

7. **接続** をクリックします。

IBM DB2 (ODBC) への接続

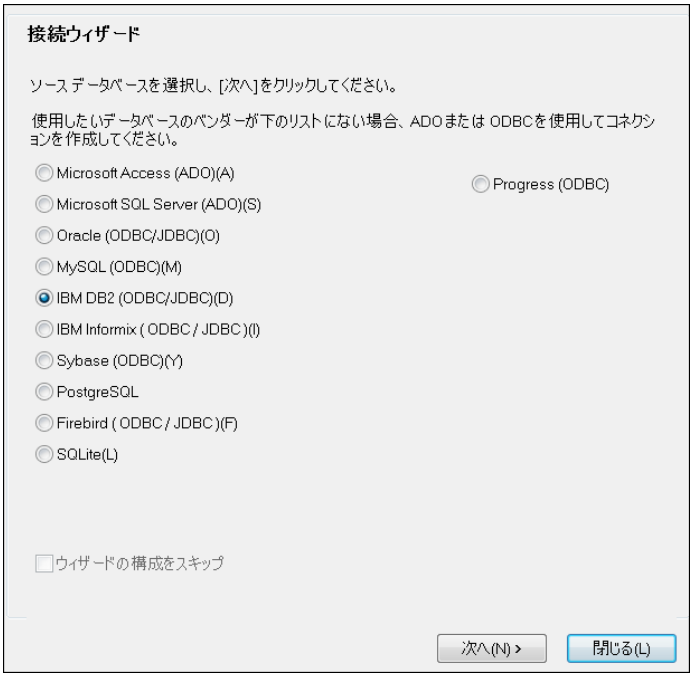
このトピックでは ODBC を使用した、IBM DB2 への接続のサンプル手順を紹介します。

前提条件：

- IBM Data Server Client は、使用中のオペレーティングシステムにインストールおよび構成される必要があります (このサンプルの場合は IBM Data Server Client 9.7 を使用します)。インストールの手順に関しては、IBM DB2 ソフトウェアにより提供されたドキュメントを確認してください。 IBM Data Server Client インストール後、使用中のコンピューターで ODBC ドライバーが使用可能か確認してください (使用可能な ODBC ドライバーの表示を参照)。
- データベースエイリアスを作成します。複数の作成方法があります：
  - IBM DB2 構成アシスタントから
  - IBM DB2 コマンドラインプロセッサーから
  - ODBC データソースウィザードから (この場合は、下に表示される手順を参照してください)
- データベース接続の詳細：ホスト、データベース、ユーザー名およびパスワード

IBM DB2 への接続：

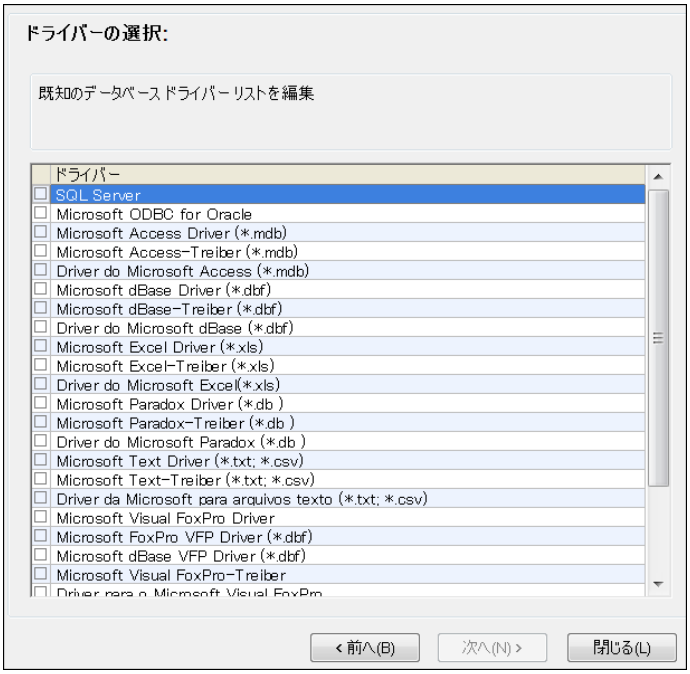
1. データベース接続ウィザードを開始します そして **IBM DB2 (ODBC/JDBC)** を選択します。



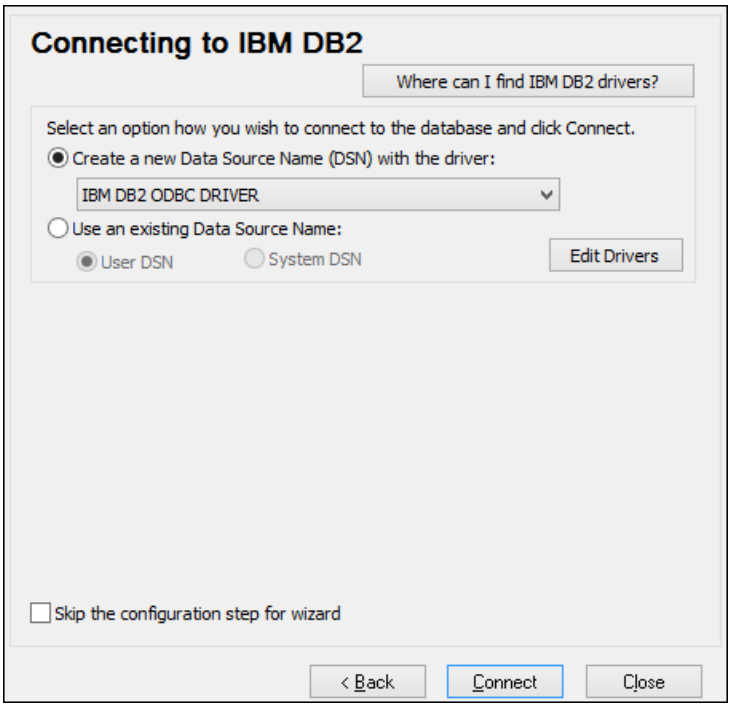
2. **次へ** をクリックします。



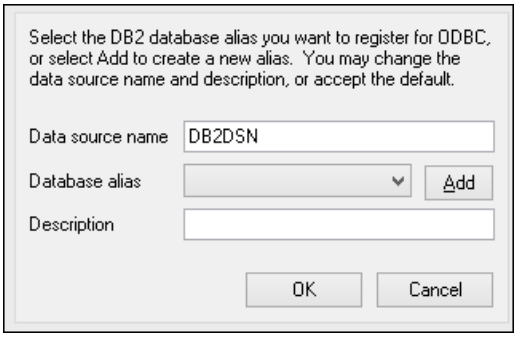
3. **ODBC** を選択して **次へ** をクリックします。データベースのための既知のリストを編集するようにプロンプトされると、IBM DB2 に適用することのできるドライバーを選択して(前提条件を参照)、**次へ** をクリックします。



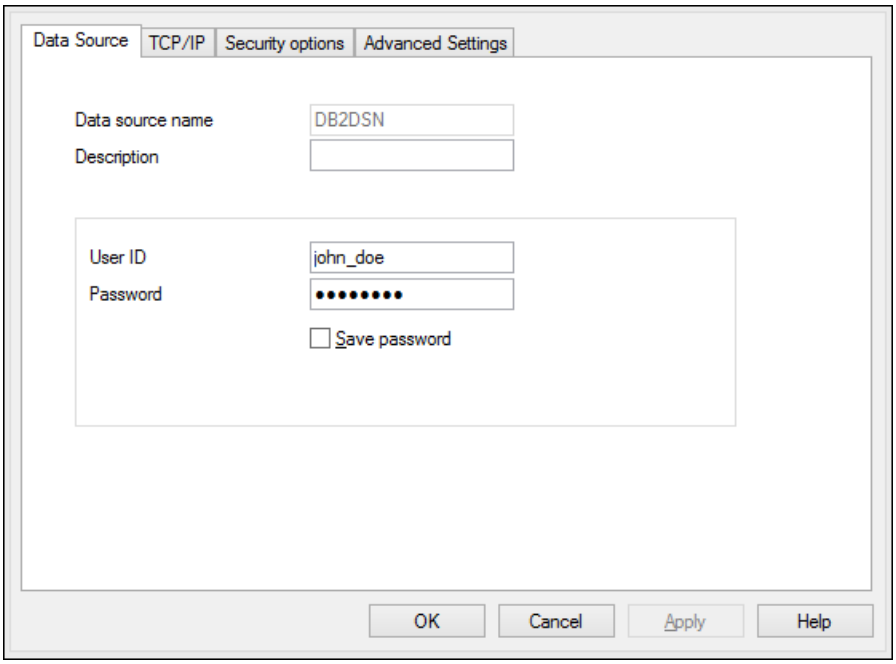
4. リストからIBM DB2 を選択して、 **接続」**をクリックします。 (ドライバーで使用可能なリストを編集するには、**[ドライバーの編集]** をクリックして、使用可能なドライバーのリストを編集します。クリックして追加または削除する IBM DB2 ドライバーをチェックまたはチェックの解除をします。ドライバーを選択または選択の解除をします。)



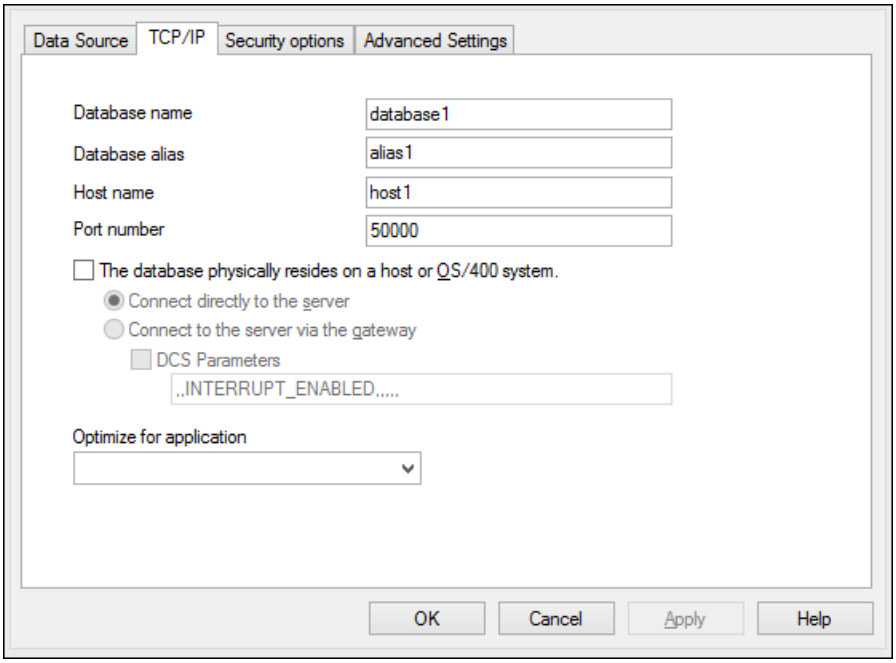
5. ソース名を入力します (このサンプルでは**DB2DSN**) **追加」**をクリックします。



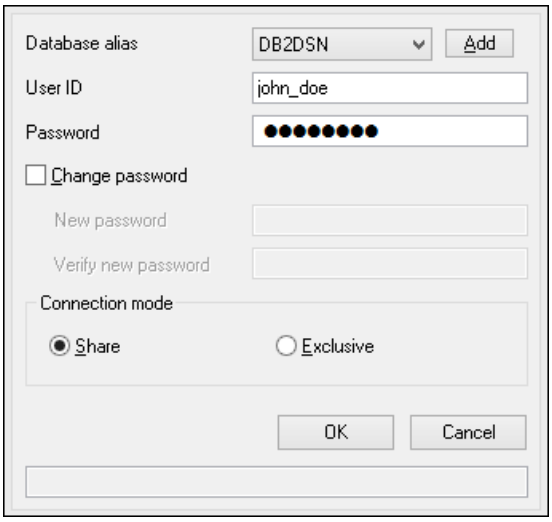
6. **[データソース]** タブで、データベースのユーザー名とパスワードを入力します。



7. **[TCP/IP]** タブで、データベース名、エイリアスの名前、ホスト名とポート名を入力して **「OK」** をクリックします。



8. ユーザ名とパスワードを再入力して「OK」をクリックします。

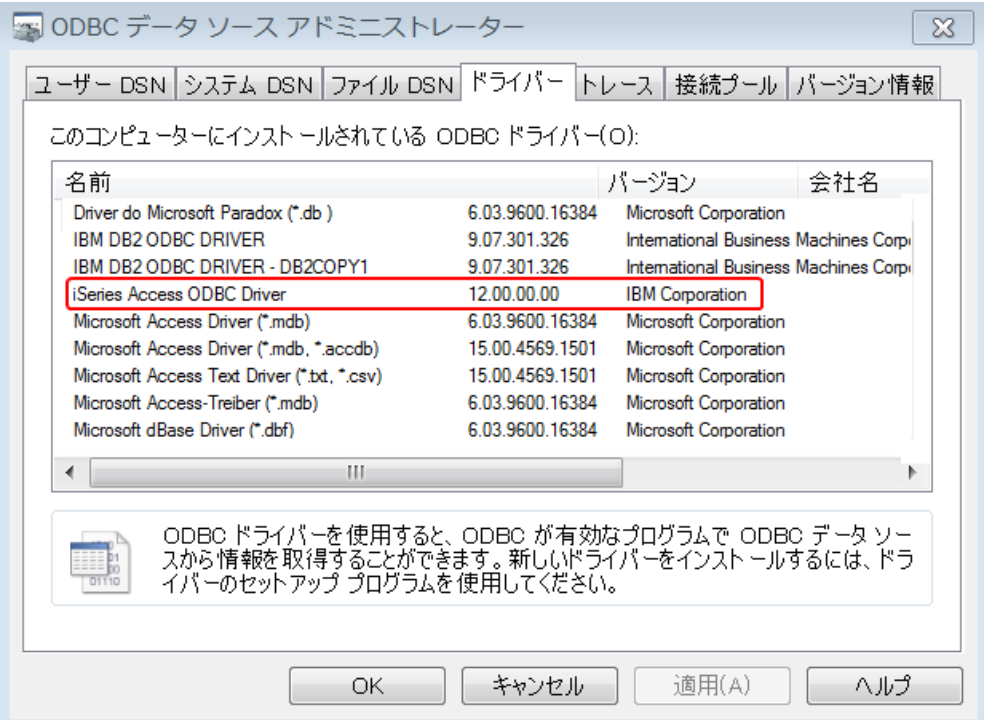


IBM DB2 for i (ODBC) への接続

このトピックは、ODBC を使用して*IBM DB2 for i* データベースへの接続のサンプル手順を説明します。

前提条件：

- *IBM System i Access for Windows* は使用中のオペレーティングシステム (このサンプルでは*IBM System i Access for Windows V6R1M0* を使用しています) にインストールされる必要があります。インストールの手順は、*IBM DB2 for i* ソフトウェアにより提供されているドキュメントを参照してください。インストール後、ODBC ドライバーが使用中のコンピューターで使用可能か確認してください (使用可能なODBC ドライバーの表示を参照)。



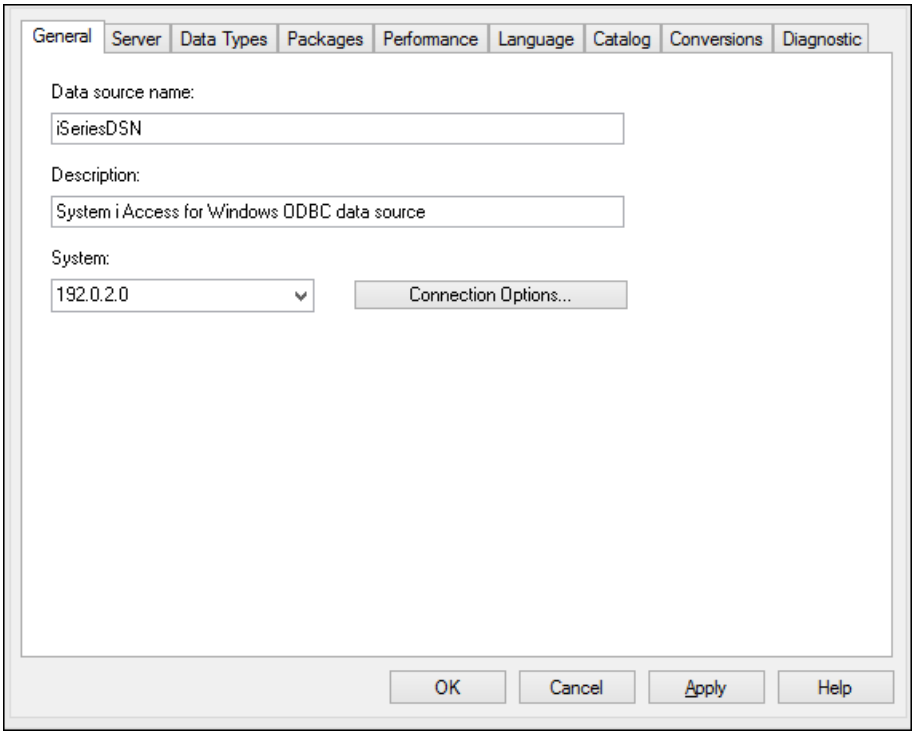
- データベース接続の詳細 データベースサーバーの I.P. アドレス データベースユーザー名、およびパスワード
- *i Navigator* を実行して、新規接続を作成するウィザードの手順を踏んで新規接続を作成してください。システムの指定をプロンプトされると、データベースサーバーのI.P. アドレスを入力してください。接続を作成した後、検証する(接続をクリックして、**ファイル** > **診断** > **接続の検証** を選択) ことが奨励されます。接続エラーが発生した場合、データベースサーバー管理者に連絡してください。

IBM DB2 for i への接続 :

1. データベース接続ウィザードを開始します.
2. **[ODBC 接続]** をクリックします。
3. **[ユーザー DSN]** をクリックします。(または、**[システム DSN]** または、後に続く手順が同じ **[ファイル DSN]** をクリックします)。
4. **[追加]** をクリックします。
5. **[iSeries Access ODBC ドライバー]** をリストから選択します。**[ユーザー DSN]** (または、該当する場合 **[システム DSN]**) をクリックします。

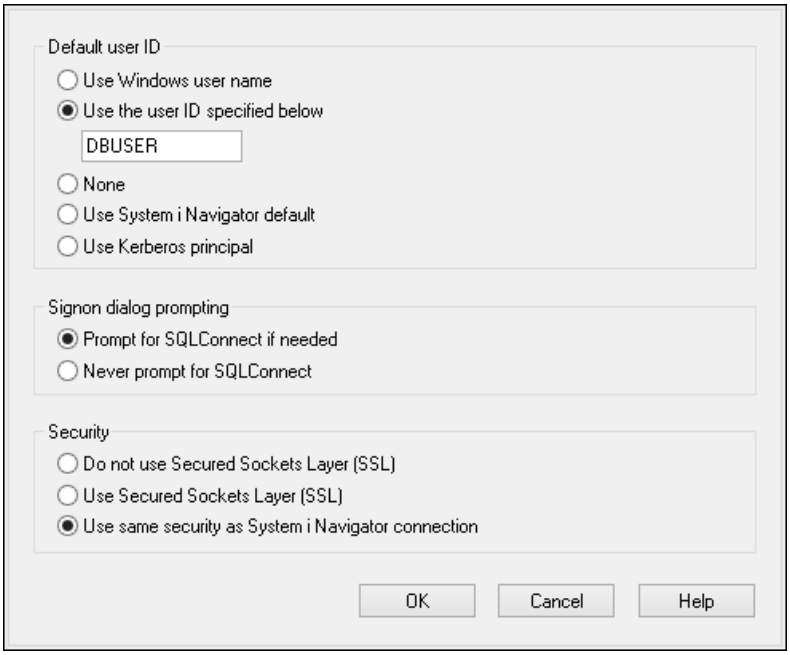


6. データソース名を入力して、システムコンボボックスから接続を選択します。このサンプルでは、データソース名は **iSeriesDSN** でシステムは **192.0.2.0** です。



メモ: ODBC *データソースを IBM DB2 for i* database に追加する場合、*クエリのタイムアウトを有効化するデフォルトのフラグが設定されます。この設定は、マッピングファイルを正確にロードするために、*MapForce *のために無効化されている必要があります。設定を無効化するには、***パフォーマンス** タブから、**詳細**をクリックして、**クエリのタイムアウトの許可** チェックボックスをクリアします。

7. **[接続オプション]** をクリックします。**[指定されたユーザー ID を使用] を選択して、**データベースユーザー (このサンプル内では **DBUSER**) の名前を入力します。



8. 「OK」をクリックします。DSN 内のリストで新しいデータソースを使用することができます。
9. 「接続」をクリックします。
10. データベースのユーザー名とパスワードをプロンプトされると入力して「OK」をクリックします。

IBM Informix (JDBC) への接続

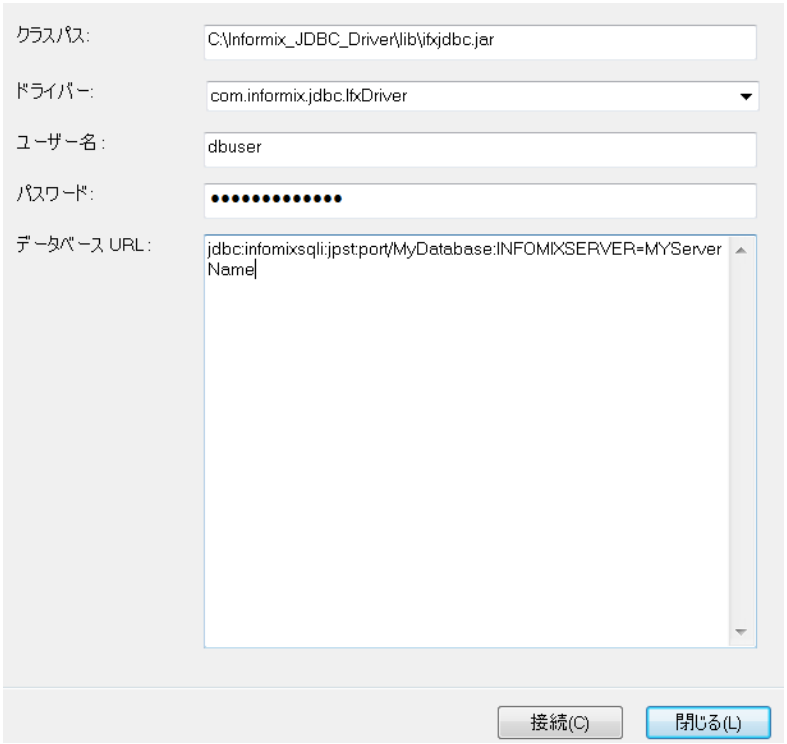
このトピックでは、JDBC を使用して IBM Informix データベースサーバーに接続する手順のサンプルを説明します。

前提条件：

- Java Runtime Environment (JRE) がオペレーティングシステムにインストールされている必要があります。
- (データベースへの接続を与える 1 つ、または、複数の jar ファイル) JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります。この例では、IBM Informix JDBC ドライバー バージョン 3.70 が使用されています。ドライバーのインストールの説明に関しては、ドライバーに付随するドキュメント、または、「IBM Informix JDBC ドライバ プログラマーズガイド」を参照してください)。
- データベース接続の詳細：ホスト、サーバーの名前、データベース、ポート ユーザー名、および パスワード

JDBC を使用して IBM Informix に接続する：

1. データベース接続ウィザードを開始します.
2. 「JDBC 接続」をクリックします。.
3. 「クラスパス」の横にデータベースへの接続を与える jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる jar ファイルは次のパスを使用して見つけることができます：**C:\Informix_JDBC_Driver\lib\ifxjdbc.jar**。jar ファイルパスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照：CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**com.informix.jdbc.IfxDriver** を選択します。このエントリは、有効な jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます（前のステップを参照してください)。



5. データベースへのユーザー名とパスワードを対応するテキストボックスに入力します。
6. ハイライトされた値を、使用するデータベースサーバーに適用する値と置き換えて、データベースURL テキストボックス内にデータベースサーバーへの接続文字列を入力します。

```
jdbc:informix-sqli://hostName:port/
databaseName:INFORMIXSERVER=myserver;
```

7. **接続** をクリックします。

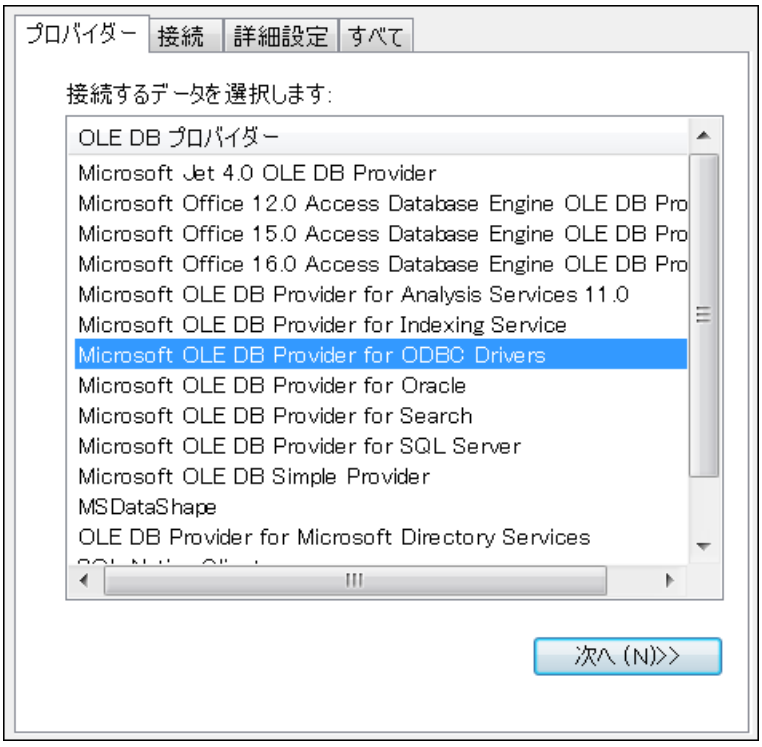
Microsoft Access (ADO) への接続

Microsoft Access データベースに接続するには、既存のMicrosoft Access データベースに接続するで説明されているとおり、データベースファイルのためのウィザードとブラウザーの手順を踏みます。別の方法としては、このトピックで説明されているとおり、明確的にADO 接続をセットアップすることです。このアプローチは、データベースがパスワードで保護されている場合、役に立ちます。

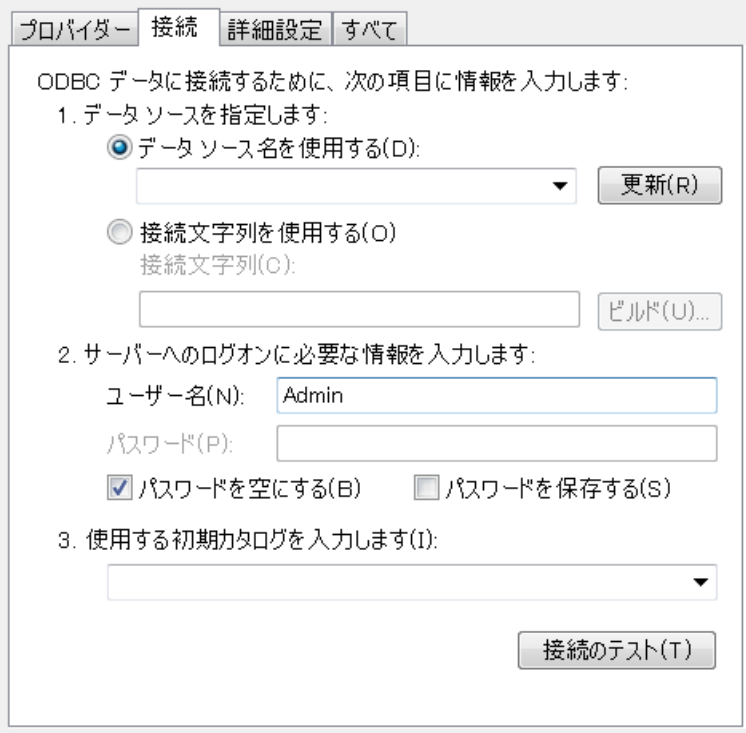
接続を使用して、に接続することもできますが、このシナリオには限度があるため、奨励されません。

パスワードで保護されているMicrosoft Access データベースに接続する:

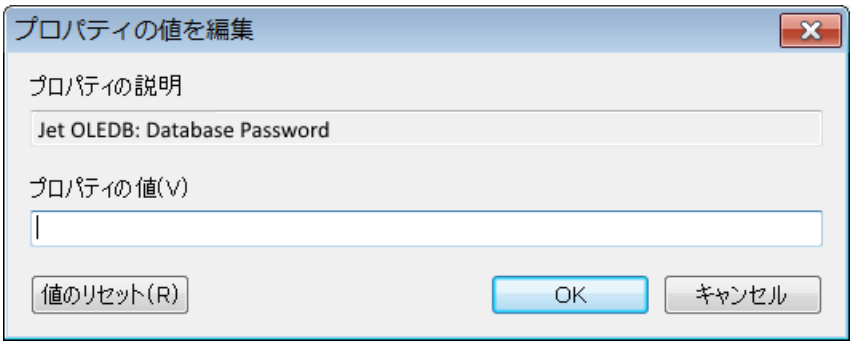
1. データベース接続ウィザードを開始します。
2. **[ADO 接続]** をクリックします。
3. **[構築]** をクリックします。



4. **[Microsoft Office 15.0 Access Database Engine OLE DB Provider]** を選択して、**次へ** をクリックします。



5. *データソースボックス*内で、Microsoft Access ファイルへのパスを入力します。ファイルがローカルネットワーク**U:\Departments\Finance\Reports\Revenue.accdb** を共有するため、ファイルをUNC フォーマットすなわち、**server1** isがサーバーの名前でdfs が共有されるネットワークの名前である、**\\server1\\dfs\Departments\Finance\Reports\Revenue.accdb** に変換します。
6. **全て** タブの**Jet OLEDB:Database Password** プロパティをダブルクリックし、データベースパスワードをプロパティの値として入力します。

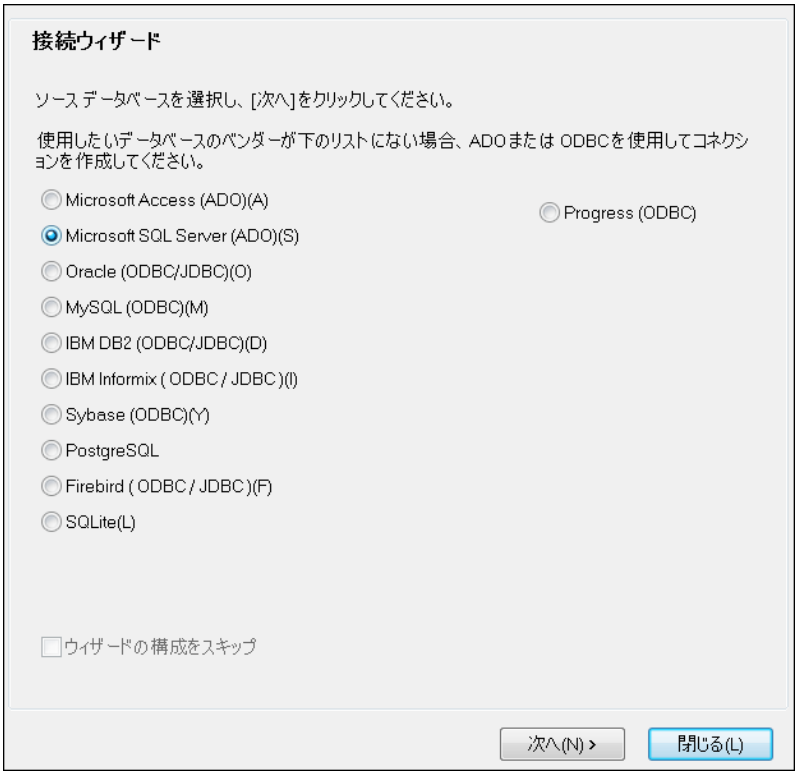


メモ: また接続できない場合、ワーキンググループ情報ファイル（**System.MDW**）を検索して、ユーザープロフィールに適用できるようにします。（手順はhttp://support.microsoft.com/kb/305542 を参照してください。）**Jet OLEDB: System database** プロパティの値を**System.MDW** ファイルのパスに設定します。

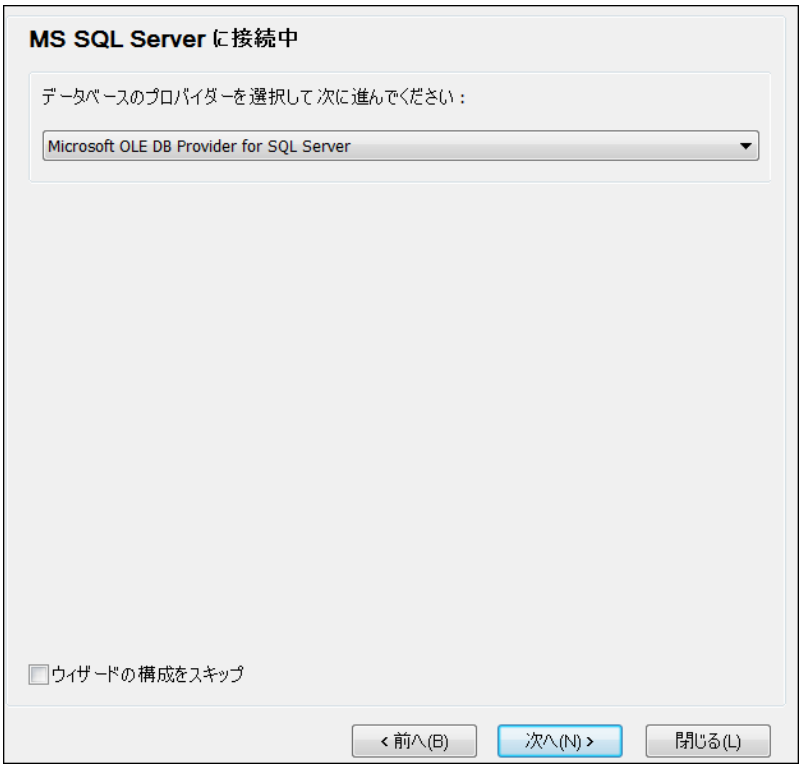
Microsoft SQL Server (ADO) への接続

Microsoft OLE DB プロバイダーを使用してのSQL Server への接続 :

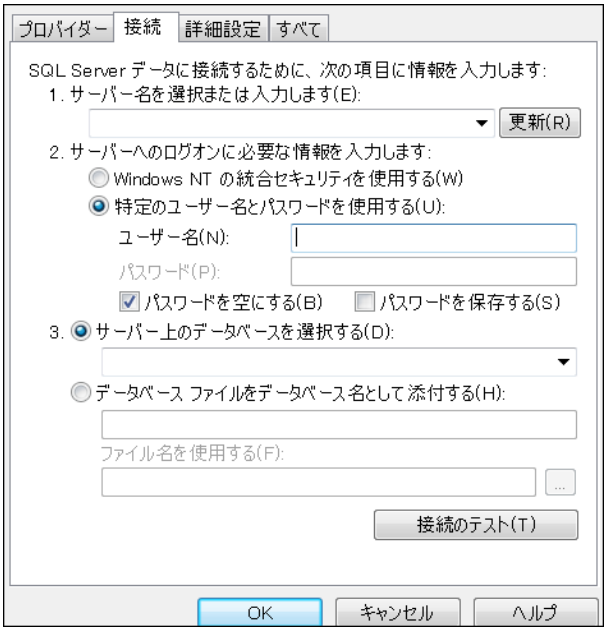
1. データベース接続ウィザードを開始します。



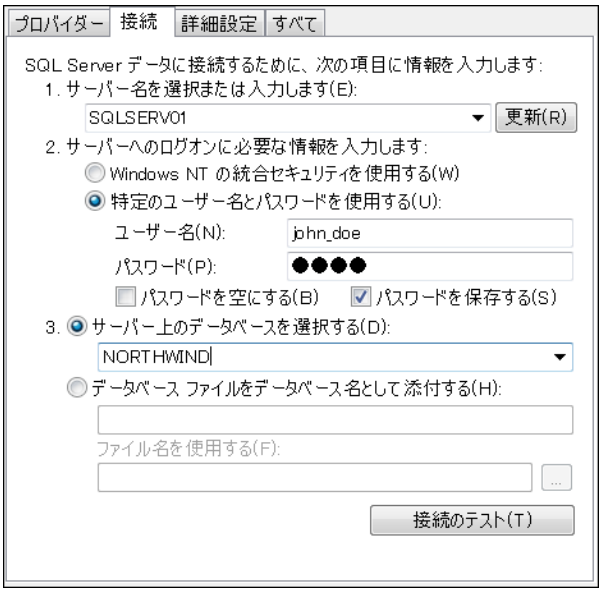
2. **Microsoft SQL Server (ADO)** を選択して、 **「次へ」** をクリックします。使用可能なADO ドライバーが表示されます。



3. **[Microsoft OLE DB Provider for SQL Server]** を選択して、 **「次へ」** をクリックします。



4. データベースサーバーの名前を選択または入力します (このサンプルでは、**SQLSERV01** です)。このネットワーク上の全てのサーバのリストを確認するには、ドロップダウンリストを展開してください。
5. データベースサーバーがWindows ドメイン上の認証されているユーザーの接続を許可している場合、**[Windows NT 統合されたセキュリティの使用]** を選択してください。それ以外の場合は、**[特定の名前とパスワードを使用する]** を選択して、関連するボックスに入力してください。
6. 接続するデータベースを選択します (このサンプルでは、**NORTHWIND** です)。
7. この時点で接続をテストするには、**[接続をテストする]** をクリックします。このステップは任意で奨励されるステップです。
8. 以下を行ってください:
   a. **[パスワードの保存を許可]** チェックボックスを選択します。
   b. **すべて** タブで、**[セキュリティ情報を保持する]** プロパティの値を**True** に変更します。



9. **「OK」** をクリックします。

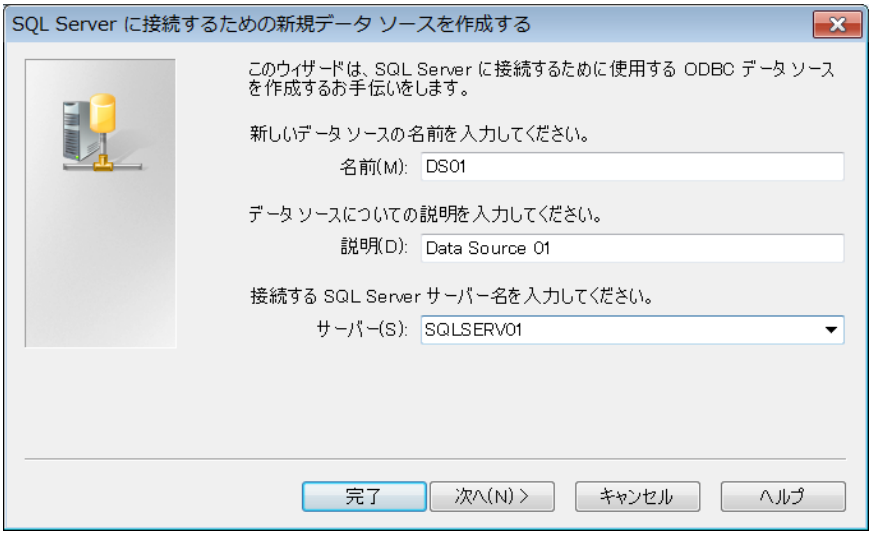
Microsoft SQL Server (ODBC) への接続

ODBC を使用してのSQL Server への接続 :

1. データベース接続ウィザードを開始します.
2. **[ODBC 接続]** をクリックします。
3. **[ユーザーDSN]** を選択して、(または管理者の特権を有する場合は、**システムDSN**) **「追加」** をクリックします。



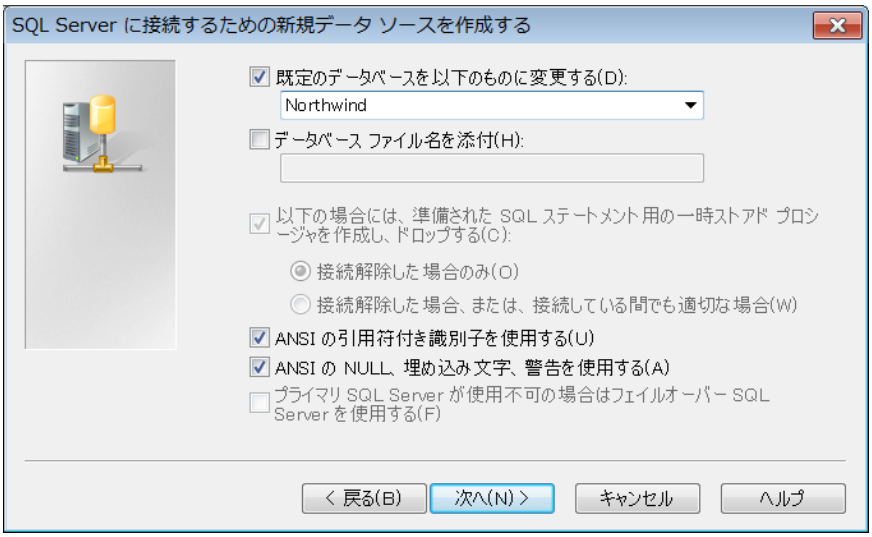
4. **[SQL Server]** (または使用可能な場合は **[SQL Server Native Client]**) を選択して、**[ユーザーDSN]** (またはシステムDSN を作成する場合は、**システム[DSN]**) をクリックします。



5. この接続を認識するための名前と詳細を入力し、接続するをリストから選択します。 (このサンプルでは、**SQLSERV01** です。)



6. データベースサーバーがWindows ドメイン上の認証されているユーザーの接続を許可している場合、**[Windows NT 承認]** を選択してください。それ以外の場合は、**[SQL Server 承認 ...]** を選択して、関連するボックスに入力してください。



1. 接続するデータベースを選択します (このサンプルでは、**NORTHWIND** です)。
7. **[完了]** をクリックします。

MySQL (ODBC) への接続

このトピックではODBC ドライバーを使用した、Windows マシンから MySQL データベースへの接続のサンプル手順を紹介します。Windows

前提条件 :

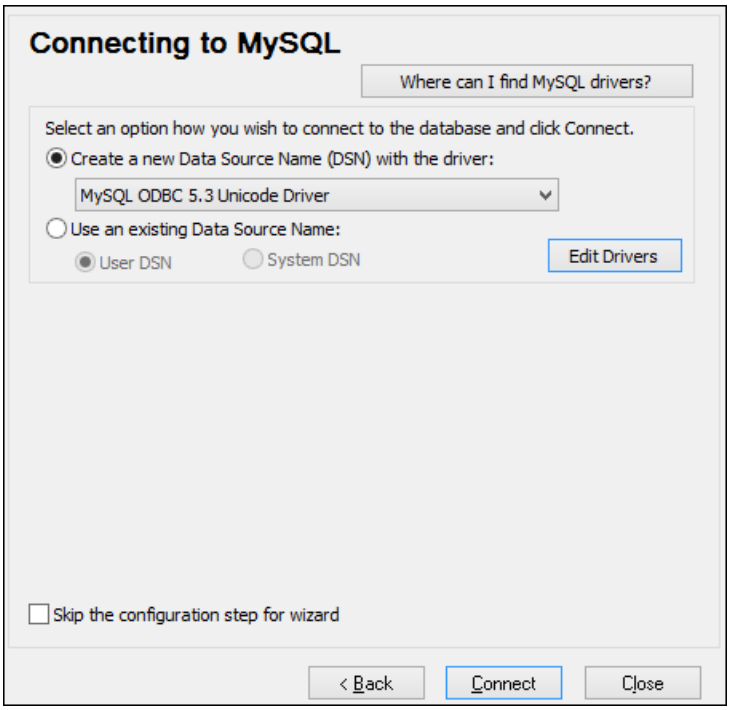
- MySQL ODBC ドライバーを使用中のオペレーティングシステムにインストールする必要があります。 (インストールに関しては、ドライバーに付随するドキュメントを参照してください)。
- データベース接続の詳細 : ホスト、データベース、ユーザー名およびパスワード

ODBC を使用してのMySQL への接続 :

1. データベース接続ウィザードを開始します。



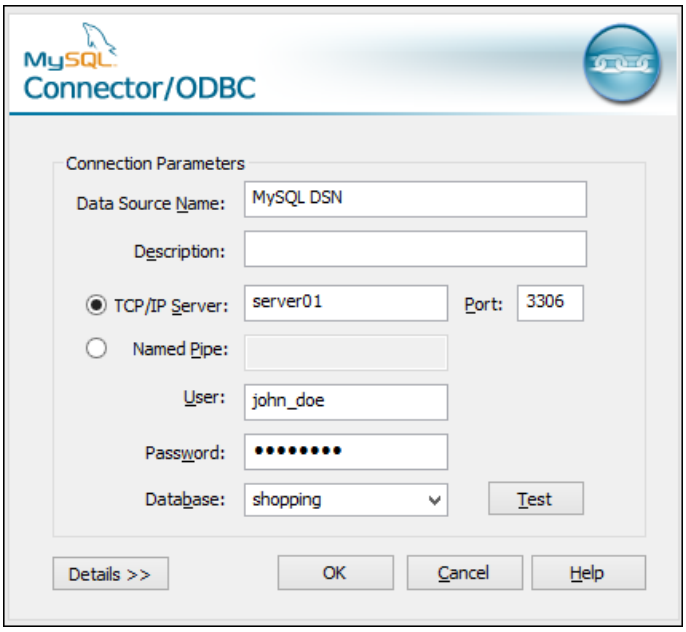
2. **[MySQL (ODBC)]** を選択して、**「次へ」**をクリックします。



3. **[ドライバーを使用して新しいデータソース名 (DSN) を作成]** を選択して、MySQL ドライバーを選択しま

す。MySQL ドライバーがリストにない場合は、**[ドライバーを編集]** をクリックして、使用可能な MySQL ドライバーを選択します (使用中のオペレーティングシステムにインストールされているすべての ODBC ドライバーがリストされています)。

4. **「接続」** をクリックします。



5. データソース名ボックスの中に、今後この ODBC データソースを認識するために役に立つ詳細を入力します。
6. データベース接続資格 (TCP/IP サーバー、ユーザー、パスワード) を入力して、データベースを選択し、 **「OK」** をクリックします。

メモ: If データベースサーバーがリモートの場合、サーバー管理者が使用中のコンピューターの IP アドレスからのリモート接続を承諾するように構成される必要があります。また、**[詳細 >>]** をクリックすると使用することのできる追加パラメーターが表示されます。デフォルトの値を変更する前に、ドライバーのドキュメントを確認してください。

Oracle (ODBC) への接続

このサンプルは、ローカルオペレーティングシステム上にインストールされている Oracle データベースクライアントを介して、ネットワークマシン上で MapForce から Oracle データベースサーバーへの接続のよく使用されている状況を説明します。

このサンプルでは、MapForce 内のデータベース接続ウィザードを使用して、ODBC データソース (DSN) のセットアップについて説明されています。既に DSN が作成されている場合は Windows 内の ODBC データソース管理者から直接作成をする事も可能です。ウィザードがプロンプトした場合はそのように選択してください。データソースに関する詳細は ODBC 接続のセットアップを参照してください。

前提条件:

- (ODBC Oracle ドライバーを含む) Oracle データベースクライアントが使用中のオペレーティングシステムでインストール、および構成されていなければなりません。 Oracle データベースクライアントのインストールおよび構成方法は Oracle ソフトウェアに付随するドキュメントを参照してください。
- Oracle ホームディレクトリにある **tnsnames.ora** ファイルは、データベース接続パラメーターを紹介するエントリを含みます。フォーマットは以下に類似します:

```
ORCL =
  (DESCRIPTION =
```

```
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = server01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = orcl)
      (SERVER = DEDICATED)
    )
  )
```

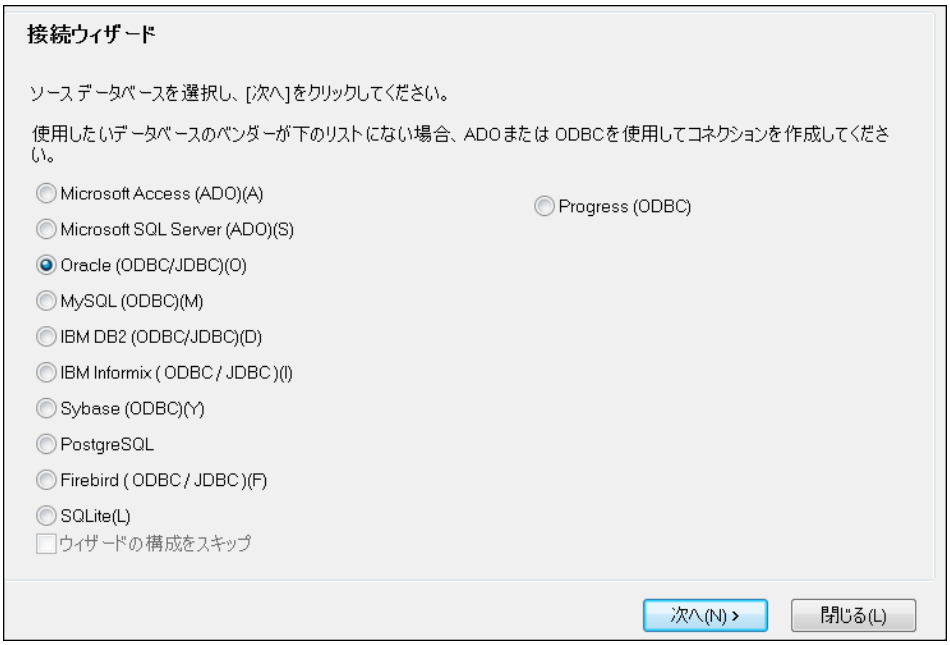
tnsnames.ora ファイルのパスはOracle ホームディレクトリがインストールされている場所によります。Oracle データベースクライアント 11.2.0 に関しては、デフォルトOracle ホームディレクトリのパスは以下の通りです：

```
C:\app\username\product\11.2.0\client_1\network\admin\tnsnames.ora
```

接続の詳細を貼り付け保存する、または(使用可能であれば)Oracle *ネット構成アシスタント*ウィザードを実行することにより **tnsnames.ora** ファイルに新規エントリを追加することができます。

ODBC を使用してのOracle への接続：

1. データベース接続ウィザードを開始します。



2. **[Oracle (ODBC / JDBC)]** を選択して、**『次へ』**をクリックします。



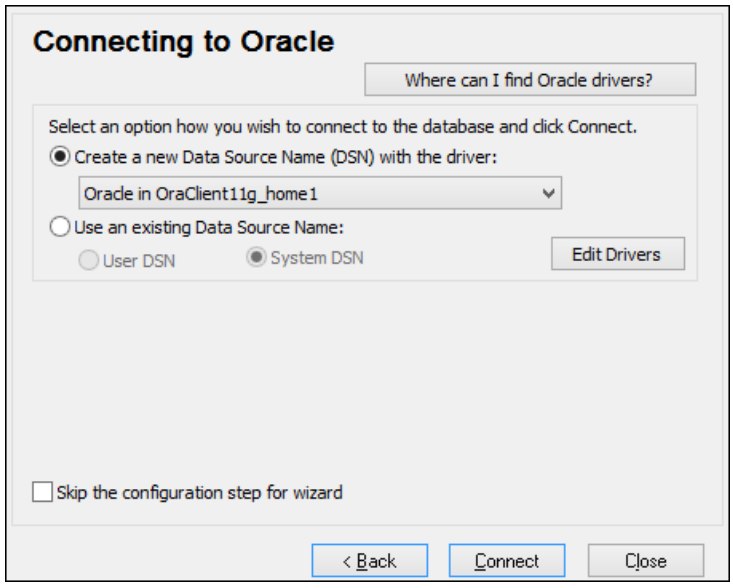
3. **[ODBC]** を選択します。



4. **[ドライバーの編集]** をクリックします。

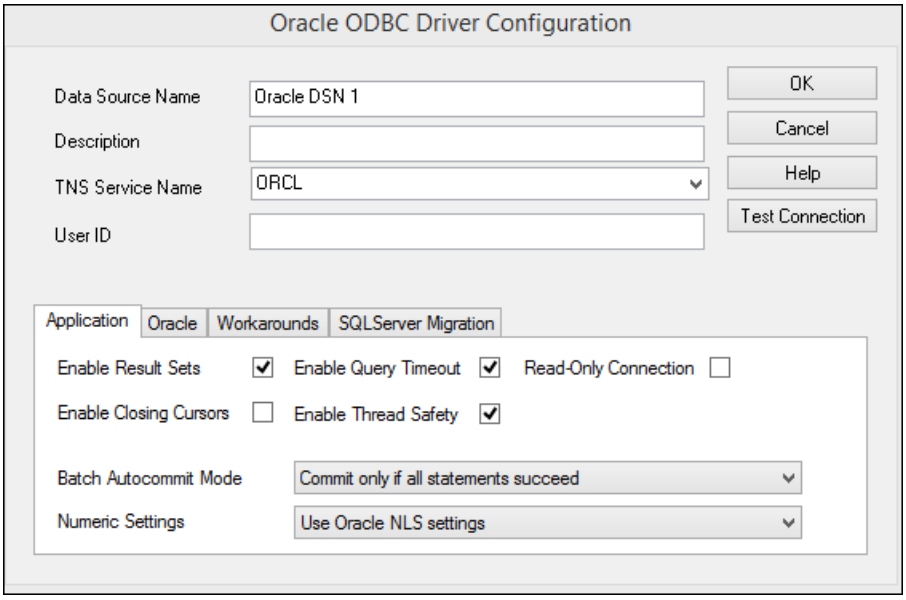


5. 使用するOracle ドライバーをせんたくします (このサンプルでは**Oracle in OraClient11g_home1** です)。Oracle クライアントをインストール後システムで使用可能なOracle ドライバーのリストを表示します。
6. **[戻る]** をクリックします。
7. **[新しいデータソース (DSN) をドライバーを使用して作成]** を選択して、ステップ 4 で選択されたOracle ドライバーを選択します。

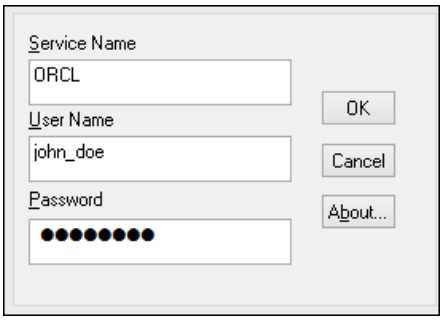


Microsoft ODBC for Oracle という名称のMicrosoft により提供されるドライバーを使用することは避けてください。Microsoft はOracle により提供されるODBC ドライバーを使用することを奨励しています。(以下参照 :http://msdn.microsoft.com/en-us/library/ms714756%28v=vs.85%29.aspx)

8. **「接続」** をクリックします。



9. データソース名テキストボックスにデータソースを識別する名前を入力します（このサンプルの場合は**Oracle DSN 1** です）。
10. TNS サービス名ボックスでは、**tnsnames.ora** ファイルで定義されている通りの接続名を入力します。前提条件を参照してください）。このサンプルでは接続名 **ORCL** はです。
11. 「**OK**」をクリックします。



12. データベースのユーザー名とパスワードを入力して、「**OK**」をクリックします。

Oracle (JDBC) への接続

JDBC インターフェイスを使用して、クライアントマシンから、Oracle データベースサーバーへの接続の方法を説明します。接続は、Oracle Web サイトから使用することのできる、**Oracle Instant Client Package (Basic)** を使用する、Java 接続として作成されます。接続の型の利点は、Oracle Instant Client Package により与えられる Java 環境と jar ライブラリのみを必要とし、更に複雑なデータベースクライアントをインストールする手間を省くことができる点です。

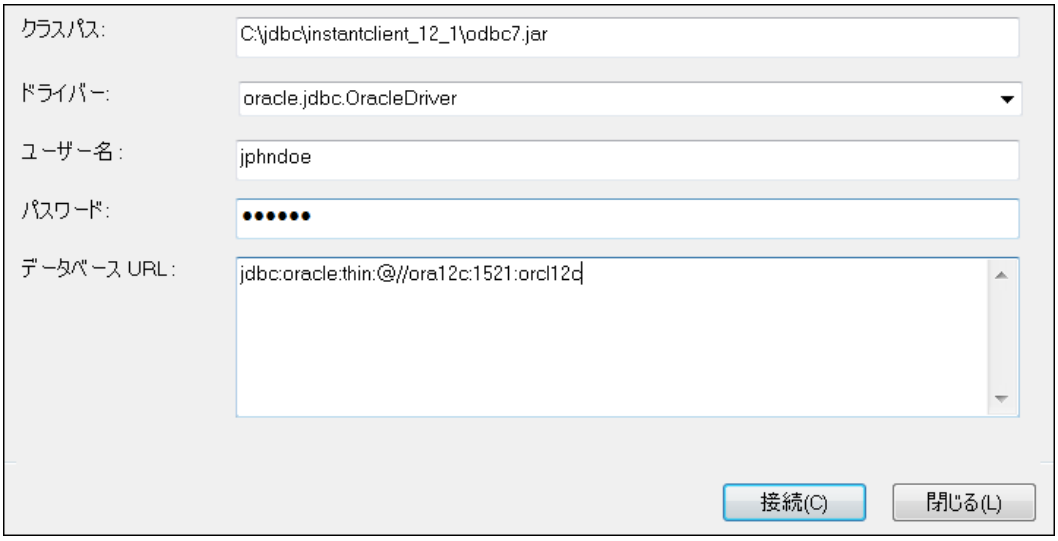
前提条件：

- Java Runtime Environment (JRE) または Java Development Kit (JDK) が使用中のオペレーティングシステムにインストールされている必要があります。
- オペレーティングシステムの PATH 環境変数は、JRE または JDK インストールの `bin` ディレクトリへのパスを含んでいる必要があります。例、`C:\Program Files (x86)\Java\jre1.8.0_51\bin`。
- **Oracle Instant Client Package (Basic)** が使用中のオペレーティングシステムで使用可能である必要が

あります。パッケージは、Oracle の公式 Web サイトからダウンロードすることができます。このサンプルは、Windows 32-ビットのための Oracle Instant Client Package バージョン 12.1.0.2.0 を使用します。
- データベース接続の詳細があります：ホスト、データベースパス、またはエイリアス、ユーザー名、およびパスワード。

クライアントパッケージを使用して Oracle に接続する：

1. データベース接続ウィザードを開始します。
2. **JDBC Connections** をクリックします。.
3. 「クラスパス」の横にデータベースへの接続を与える jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる jar ファイルは次のパスを使用して見つけることができます：**C:\jdbc\instantclient_12_1\odbc7.jar**。jar ファイルパスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照：CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**oracle.jdbc.OracleDriver** または **oracle.jdbc.driver.OracleDriver** を選択します。このエントリは、有効な jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます（前のステップを参照してください）。
5. データベースへのユーザー名とパスワードを対応するテキストボックスに入力します。



6. ハイライトされた値を、使用するデータベースサーバーに適用する値と置き換えて、データベース URL テキストボックス内にデータベースサーバーへの接続文字列を入力します。

```
jdbc:oracle:thin:@//host:port:service
```

7. **接続** をクリックします。

PostgreSQL (ODBC) への接続

このトピックでは ODBC ドライバーを使用した、Windows マシンから PostgreSQL データベースへの接続のサンプル手順を紹介します。Windows からは、PostgreSQL ODBC ドライバーを入手することができませんので、個別にダウンロードおよびインストールする必要があります。このサンプルでは、公式 Web サイトからダウンロードされた psqlODBC ドライバー (バージョン 09_03_300-1) を使用しています。（データベースドライバー概要も参照してください）。

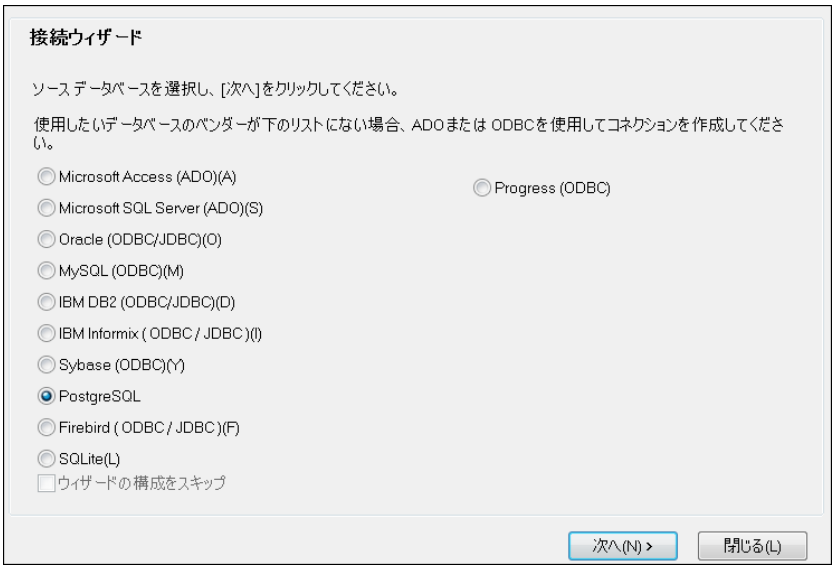
メモ： (ODBC ドライバー無しで) 直接 PostgreSQL データベースサーバーに接続するには、次を参照してください：PostgreSQL 接続のセットアップ.

前提条件：

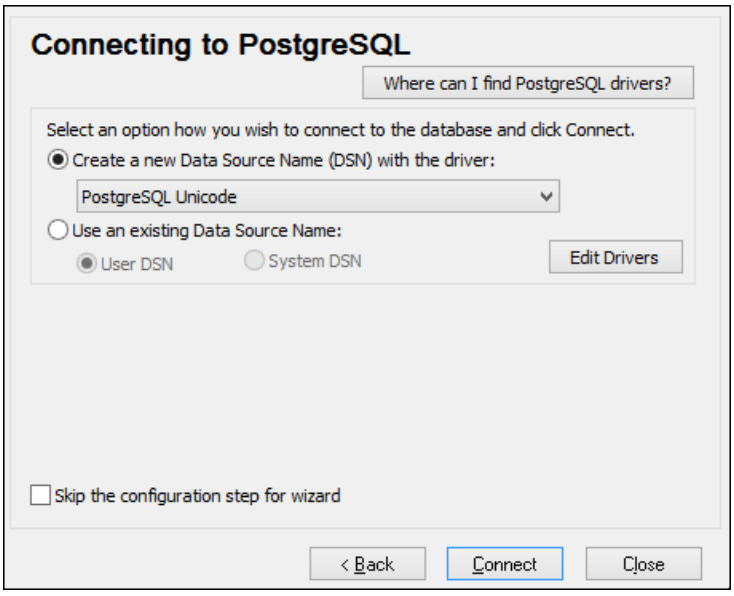
- *psqlODBC* ドライバーを使用中のオペレーティングシステムにインストールする必要があります。（インストールに関しては、ドライバーに付随するドキュメントを参照してください）。
- データベース接続の詳細：ホスト、データベース、ユーザー名およびパスワード

ODBC を使用してのPostgreSQL への接続：

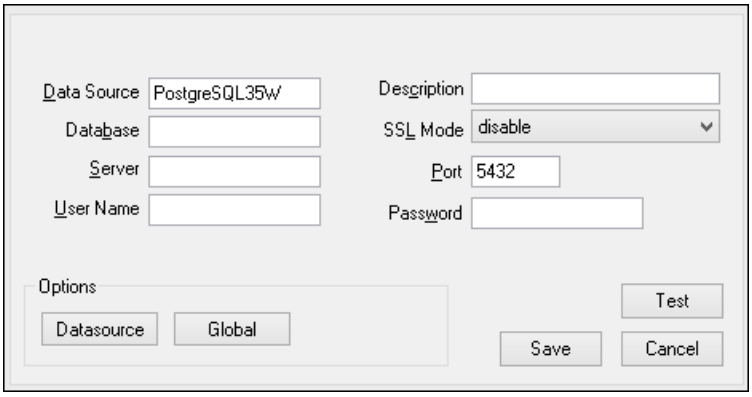
1. データベース接続ウィザードを開始します。



2. **PostgreSQL (ODBC)** を選択して、**「次へ」**をクリックします。



3. **[ドライバーを使用して新しいデータソース名（DSN）を作成]** を選択して、PostgreSQL ドライバーを選択します。MySQL ドライバーがリストにない場合は、**[ドライバーを編集]** をクリックして、使用可能な PostgreSQL ドライバーを選択します（使用中のオペレーティングシステムにインストールされているすべての ODBC ドライバーがリストされています）。
4. **「接続」**をクリックします。



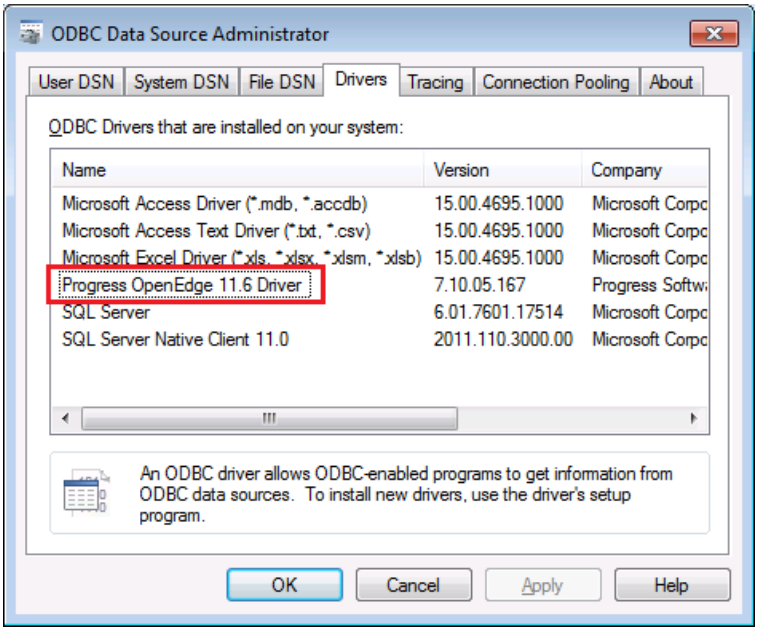
5. データベース接続資格 (データベース、サーバー、ポート、ユーザー名、パスワード) を入力して、 「OK」 をクリックします。

Progress OpenEdge (ODBC) への接続

このトピックは、Progress OpenEdge 11.6 ODBC ドライバーを使用して、Progress OpenEdge データベースサーバーへの接続の簡単な手順について説明しています。

前提条件

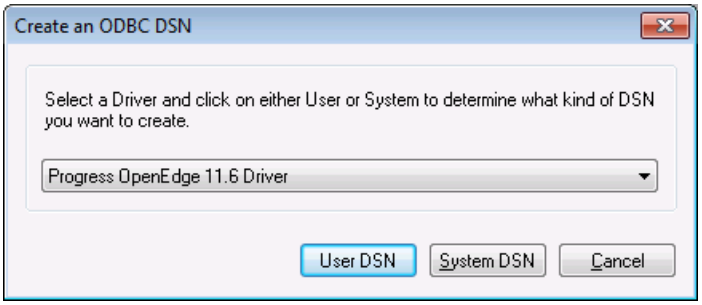
- *Progress OpenEdge* ドライバーのための *ODBC* コネクタが使用中のオペレーティングシステムにインストールされている必要があります。Progress OpenEdge ODBC ドライバーは、ベンダーの Web サイトからダウンロードすることができます (次も参照 : データベース ドライバーの概要)。MapForce の 32 ビットバージョンを作動する場合は、32 ビット ドライバーをダウンロードし、64 ビットバージョンを作動する場合は、64 ビット ドライバーをダウンロードしてください。インストールの後、ODBC ドライバーが使用中のマシンで使用することができるかを確認してください (次も参照 : 使用することのできる ODBC ドライバーを確認する)。



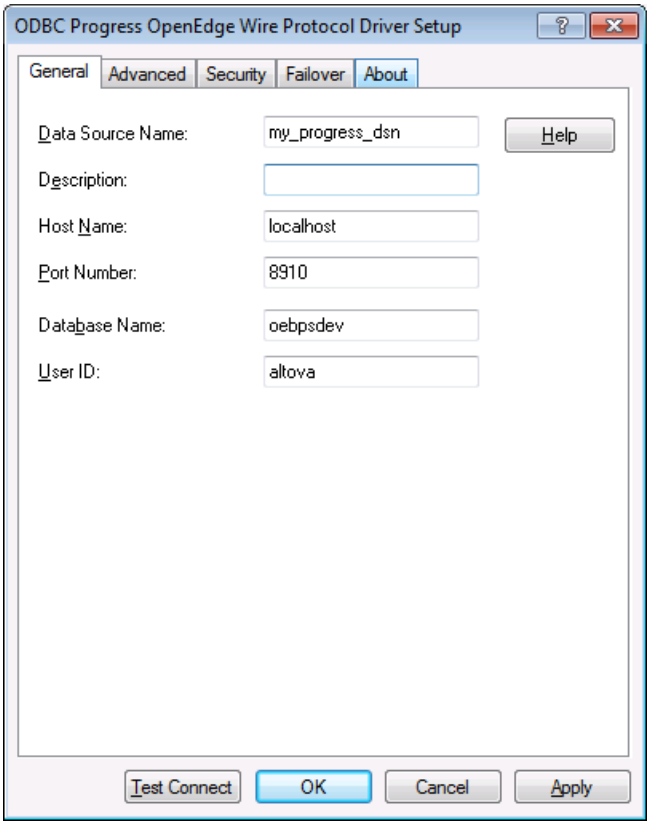
- データベース接続の詳細があります : ホスト名、ポート番号、データベース名、ユーザー ID、および パスワード。

ODBC を使用して Progress OpenEdge に接続する

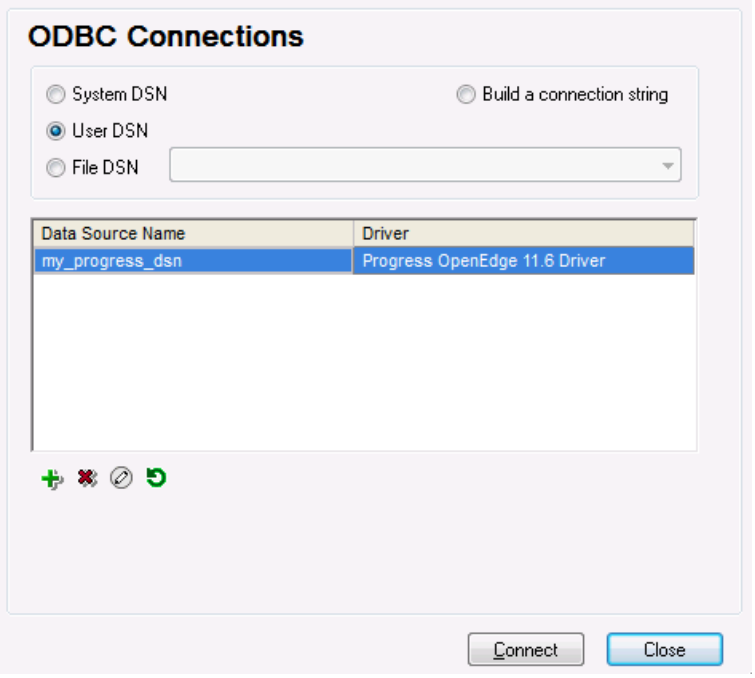
1. データベース接続ウィザードを開始します.
2. **ODBC Connections**をクリックします。
3. **「ユーザーDSN」**をクリックします。(または、次の手順が類似している**「システムDSN」**、または、**「ファイルDSN」**をクリックします)。
4. **「追加」**をクリックします。
5. **「Progress OpenEdge ドライバー」**をリストから選択し、**「ユーザーDSN」**(または、適用できる場合は、**「システムDSN」**)をクリックします。



6. データベース接続の資格情報(データベース、サーバー、ポート、ユーザー名、パスワード)を入力し、**「OK」**をクリックします。入力されたデータを保存する前に、接続を検証するには、**「接続をテストする」**をクリックします。



7. **「OK」**をクリックします。新規データソースがODBC データソースのリストに表示されます。



8. **接続** をクリックします。

Progress OpenEdge (JDBC) への接続

JDBC を使用して Progress OpenEdge 11.6 データベースサーバーへの接続するためのサンプルの手順が説明されています。

前提条件

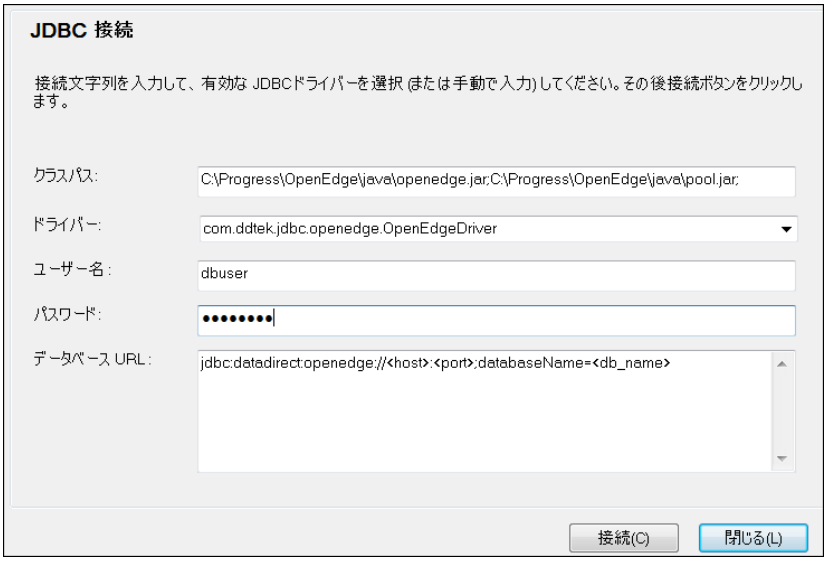
- Java Runtime Environment (JRE) または Java Development Kit (JDK) が使用中のオペレーティングシステムにインストールされている必要があります。 MapForce (32-ビット、 64-ビット) のプラットフォームが、JRE/JDK に一致していることを確認してください。
- オペレーティングシステムの PATH 環境変数は、JRE または JDK インストールの bin ディレクトリへのパスを含んでいる必要があります。例、`C:\Program Files (x86)\Java\jre1.8.0_51\bin`.
- Progress OpenEdge JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります。このサンプルでは、JDBC 接続は、**C:\Progress\OpenEdge\java** に OpenEdge SDK のインストールの一部として存在する **openedge.jar** と **pool.jar** ドライバーコンポーネントファイルにより与えられています。 .
- データベース接続の詳細があります :ホスト、データベースパス、またはエイリアス、ユーザー名、およびパスワード。

JDBC を使用して OpenEdge へ接続する

1. データベース接続ウィザードを開始します.
2. **「JDBC 接続」** をクリックします。.
3. 「クラスパス」の横にデータベースへの接続を与える jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた jar ファイルパスのリストを入力することもできます。 このサンプルでは、必要とされる jar ファイルは次のパスを使用して見つけることができます **`C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;`**。 jar ファイルパスをオペレーティングシステムの

CLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照 :CLASSPATH の構成)。

4. 「ドライバー」ボックスから、**com.ddtek.jdbc.openedge.OpenEdgeDriver** を選択します。このエントリは、有効な jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。



5. データベースへのユーザー名とパスワードを対応するテキストボックスに入力します。
6. ハイライトされた値を、使用するデータベースサーバーに適用する値と置き換えて、データベースURL テキストボックス内にデータベースサーバーへの接続文字列を入力します。

```
jdbc:datadirect:openedge://host:port;databaseName=db_name
```

7. **「接続」**をクリックします。

Sybase (JDBC) への接続

このトピックでは、JDBC を使用して Sybase データベースサーバーに接続する手順のサンプルを説明します。

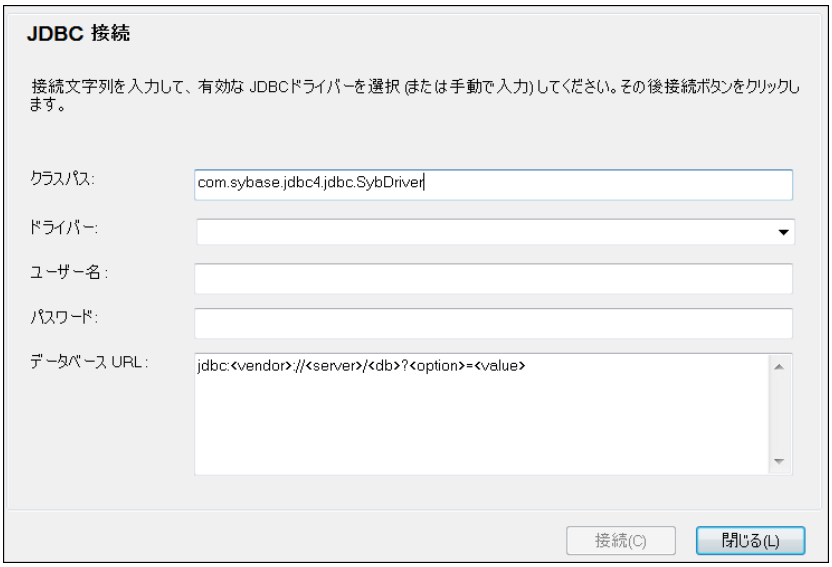
前提条件 :

- Java Runtime Environment (JRE) がオペレーティングシステムにインストールされている必要があります。
- Sybase *jConnect* ドライバーがオペレーティングシステムにインストールされている必要があります (このサンプルでは、*Sybase Adaptive Server Enterprise PC Client* インストールの一部としてインストールされている *jConnect 7.0* が、使用されています)。ドライバーのインストールの手順に関しては、ドライバーに付随するドキュメント、または Sybase のドキュメントを参照してください)。
- データベース接続の詳細 :ホスト、サーバーの名前、データベース、ポート、ユーザー名、およびパスワード

JDBC を使用して Sybase へ接続する :

1. データベース接続ウィザードを開始します。
2. **「JDBC 接続」**をクリックします。
3. 「クラスパス」の横にデータベースへの接続を与える jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる jar ファイルは次のパスを使用して見つけることができます :**C:\sybase\jConnect-7_0\classes\jconn4.jar**。jar ファイルパスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照 :CLASSPATH の構成)。

4. 『ドライバー」ボックスから、 を選択します。 『ドライバー」ボックスから **com.sybase.jdbc4.jdbc.SybDriver** を選択します。このエントリは、有効な jar ファイルパスが、『クラスパス」テキストボックス内、または、オペレーティングシステムのCLASSPATH 環境変数内で検出された場合のみ使用することができます（前のステップを参照してください）。



5. データベースのユーザー名とパスワードを対応するテキストボックスに入力します。
6. ハイライトされた値をデータベースサーバーに適用することのできる値と置き換えて。データベースURL テキストボックス内のデータベースサーバーの接続文字列を入力します。

```
jdbc:sybase:Tds:hostName:port/databaseName
```

7. **接続」**をクリックします。

7.2.2 Linux と Mac上でのデータベース接続

以下のAltova サーバー製品にライセンスが与えられている場合？MobileTogether Server、MapForce Server またはStyleVision Server、よくあるシナリオとして考えられるのは、MobileTogether デザインのデザイン、MapForce マッピング、またはStyleVision 変換を Windows デスクトップコンピューターでおこない、実行を自動化するためにサーバーマシン(Windows、Linux、またはMac) にデプロイすることです。

このドキュメントでは、[サーバー実行ファイル] は以下のファイルの種類を意味します：

- MapForce Server 実行ファイル (mfx)
- MobileTogether デザインファイル (mtd)
- Portable XML Forms (.pxf) としてのStyleVision 変換 (sps) パッケージ

サーバー実行ファイルをデプロイする時以下のシナリオが考えられます：

1. **"Windows上でのデザインと実行 "**。 このシナリオでは、MobileTogether デザインのデザイン、MapForce マッピング、StyleVision 変換をWindows 上で行い、対応するサーバ実行ファイルも Windows システムで実行します。(同じWindows マシンまたは、リモートのWindows サーバーで行うことができます)。
2. **"Windows 上でデザイン、Linux またはMac上で実行 "**。このシナリオでは、上記のファイルのデザインを Windows 上で行い、対応するサーバー実行ファイルをLinux およびMac に実行するためにデプロイします。

"Windows上でのデザインと実行"シナリオでは、使用可能なデータベーステクノロジーとドライバーはADO、ODBC、JDBC およびSQLite 接続（データベースドライバー概要を参照）から構成されます。

"Windows 上でデザイン、Linux または Mac上で実行"シナリオでは、ADO およびODBC 接続はサポートされていません。このシナリオでは、直接的なSQLite 接続（SQLite 接続を参照）とJDBC 接続（JDBC 接続を参照）がサポートされます。

サーバーにサーバー実行ファイルをデプロイすると、データベースはデプロイされるパッケージに含まれておらず、（SQLite およびMicrosoft Access等のファイルベースのデータベースでも同様）デプロイサーバー側でも接続がセットアップされていなければなりません。すなわち、同じデータベース構成がデザインを行うオペレーティングシステムとファイルをデプロイするサーバー側で行われていなければならないのです。このルールに対しての例外は、（ドライバーをベースにしない）ネイティブのPostgreSQL 接続です。ネイティブのPostgreSQL 接続は、MapForce の外部の構成を必要としません。詳細に関しては、次を参照してください PostgreSQL 接続のセットアップ。

通常、双方のマシンで同じデータベースの構成が必要なため、サーバー実行ファイルを異なるオペレーティングシステムにデプロイするシナリオは若干複雑です。デザインをローカルで行い、デプロイをリモートで行う複雑性をバイパスするため、MapForce、MobileTogether Designer および StyleVision で使用可能なグローバルリソース機能を使用することを検討してください。

例えば、異なる 2 つのグローバルリソース構成を定義し、同じデータベースに接続することができます：1 つはWindows スタイルのパス規約を使用して接続設定を行い、もう 1 つは、Linux スタイルのパス規約を使用します。最初の接続を使用して、デザイン中にファイルのテストを行い、第 2 の接続をLinux サーバー上でのファイルの実行に使用することができます。

7.2.2.1 Linux とMac 上での SQLite 接続

Altova サーバー製品には、サポートが統合されているため、Linux とMac にSQLite を別途にインストールする必要はありません。ですから、サーバー実行ファイルは、SQLite データベースへの呼び出しを含み、SQLite を最初にインストールすることなく実行することができます。ですが、サーバー実行ファイルがLinux またはMac マシン上でデータベースファイルへの正しいパスを使用していることを確認する必要があります。すなわち、Linux またはMac サーバーでサーバー実行ファイルを実行する前に、SQLite データベースファイルがPOSIX (Portable Operating System Interface) 準拠であるパスを介して参照されていることを確認してください。これは、Windows スタイルではないのドライブ文字がパスに使用されていないこと、また、ディレクトリがスラッシュ（/）で区切られていることを前提にしています。例えば、パス **/usr/local/mydatabase.db** はPOSIX 準拠で、パス **c:\sqlite\mydatabase.db** は準拠ではありません。

Altova 製品の外部で、SQLite データベースと作業するためにシェルが必要な場合、（例えば、データベースの作成など）、コマンドラインシェルを公式のWeb サイト（www.sqlite.org）から、または第三者の代替よりダウンロードする必要があるかもしれません。

7.2.2.2 Linux とMac 上での JDBC 接続

Linux またはMac 上でのJDBC 接続のセットアップ：

1. データベースベンダーにより提供されるJDBC ドライバーをダウンロードして、オペレーティングシステムにインストールします。オペレーティングシステムが32 ビットで作動する場合は32 ビット版バージョンを、オペレーティングシステムが64-ビットで作動する場合は 64 ビット版バージョンを選択してください。
2. 環境変数をJDBC ドライバーがインストールされている場所に設定してください。通常、CLASSPATH 変数と必要があればその他複数を設定します。どの特定の環境変数が構成されるかを検出するには、JDBC ドライバーに付随するドキュメントを確認してください。

メモ： Mac OS では、システムはJDBC ライブラリが **/Library/Java/Extensions** ディレクトリにあることを必要とし

ます。ですから、JDBC ドライバーをこの場所でアンパックすることが奨励されます。それ以外の場合、システムを構成し、JDBC ライブラリがJDBC ドライバーをインストールしたパスを検索するようにする必要があります。

7.2.2.3 Mac OS X Yosemite 上での Oracle 接続

Mac OS X Yosemite 上では、**Oracle Database Instant Client** を介してOracle データベースに接続することができます。Java 8 以前のJava バージョンのMac OS では**JDBC Thin for All Platforms** ライブラリを介して接続することもできます。この点に関しての手順はこのトピックでは、無視することができます。

Oracle Instant Client をOracle の公式ダウンロードページからダウンロードすることができます。Oracle ダウンロードページでは複数のInstant Client パッケージがあることに注意してください。Oracle Call Interface (OCI) サポートのあるパッケージが選択されていることを確認してください (例えばInstant Client Basic)。また、オペレーティングシステムが32-ビットで作動する場合は32 ビット版バージョンを、オペレーティングシステムが 64 ビットで作動する場合は 64 ビット版バージョンを選択してください。

ダウンロードした後、Oracle Instant Client をアンパックし、インストーラと共にパックされているプロパティリストファイル (.plist) を編集すると、以下の環境変数が対応するドライバーパスをポイントします。例えば:

変数	サンプル値
CLASSPATH	/opt/oracle/instantclient_11_2/ojdbc6.jar:/opt/oracle/instantclient_11_2/ojdbc5.jar
TNS_ADMIN	/opt/oracle/NETWORK_ADMIN
ORACLE_HOME	/opt/oracle/instantclient_11_2
DYLD_LIBRARY_PATH	/opt/oracle/instantclient_11_2
PATH	$PATH:/opt/oracle/instantclient_11_2

メモ: 使用中のオペレーティングシステムでOracle Instant Client ファイルがインストールされている場所で上記のサンプル値を編集してください。

7.2.3 データベースマッピングの概要

このセクションはMapForce 内出データベースと作業する方法に関する概要です。接続の確立に成功した後、どのようにしてデータベースと作業するかについて説明されています (次を参照 :データベースへの接続)。これには、マッピングに表示されるデータベースオブジェクト、データベースリレーションシップの処理、マッピングプロセスに適用することのできるデータベースの構成 などが含まれます。データベースコンポーネントから、または、データベースコンポーネントにデータをマッピングする際の特定の目的を達成する方法もこのセクションの例に含まれています。

7.2.3.1 マッピングにデータベースを追加する

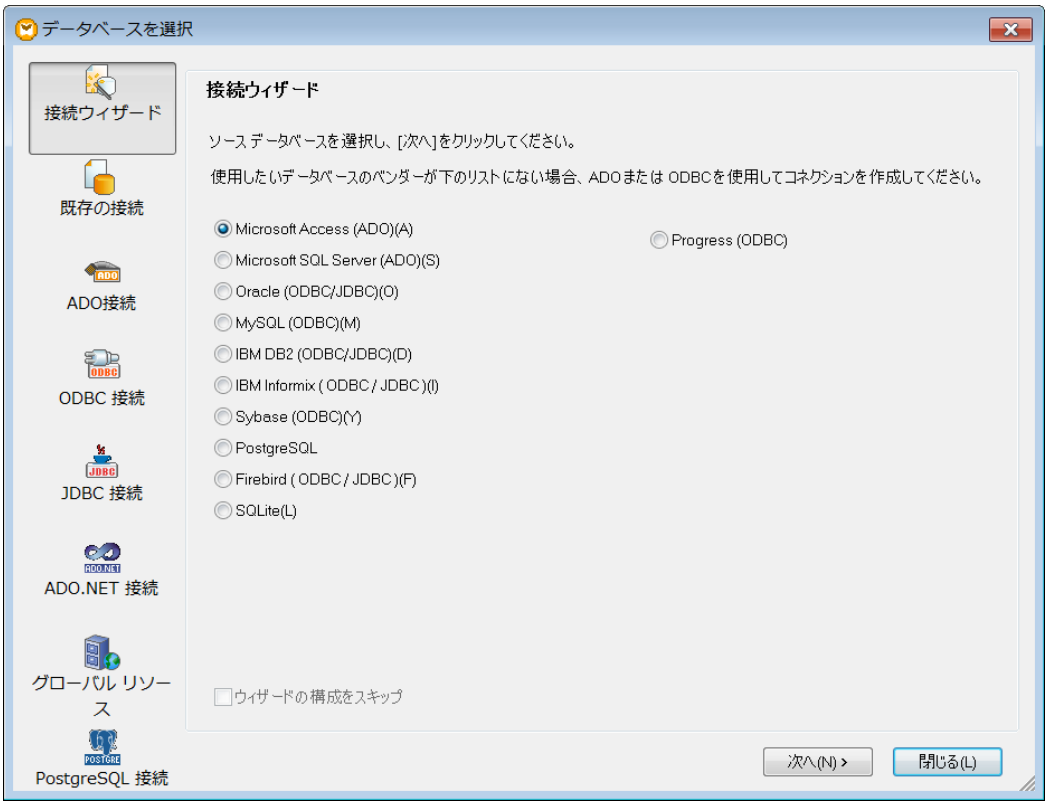
マッピングにデータベースを追加する前に、データベースマッピングがサポートされている変換言語が選択されていることを確認してください。これは、BUILT-IN 変換言語、または次の言語であることができます :C++、C#、Java (次も参照してください:変換言語の選択)。マッピングをFlowForce Server にデプロイ、または、MapForce Server で実行、または、バルク転送とストアドプロシージャなどの機能を使用する場合、BUILT-IN が変換言語として選択される必要があります。

希望する変換言語が選択されると、データベースをマッピングに次の方法で追加することができます :

- **挿入** メニューから **データベース** をクリックします。

- **データベースの挿入**（ ）ツールバーボタンをクリックします。

これらのアクションを取ると、データベースへの接続に必要なステップを導くデータベース接続ウィザードが表示されます。



データベース接続ウィザード

メモ: 高度なシナリオの一部では、データベースに変数として追加することができます（次を参照：変数の使用）。データベース構成を変数として追加する場合は、上記と同じデータベース接続ウィザードが表示されます。

このウィザードを使用してMapForce によりサポートされているデータベースへの接続の方法に関しては、以下を参照してください：データベースへ接続 また、特に、以下を含む順序を追った例：

- Firebird への接続 (ODBC)
- IBM DB2 への接続 (ODBC)
- 既存のMicrosoft Access Database への接続
- Microsoft SQL Server への接続 (ADO)
- MySQL への接続 (ODBC)
- Oracle への接続 (ODBC)
- PostgreSQL への接続 (ODBC)
- Progress OpenEdge への接続 (JDBC)
- 既存のSQLite Database への接続
- Sybase への接続 (JDBC)

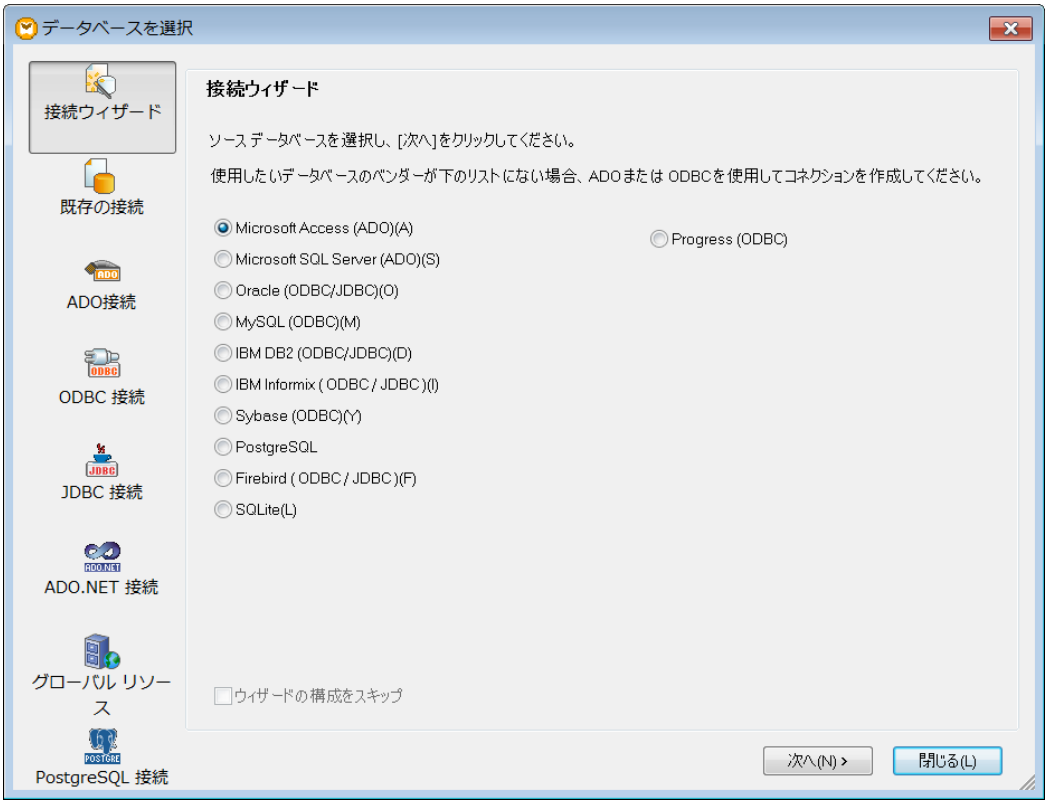
データベースへの接続に成功すると、マッピングに表示されるデータベースオブジェクトを選択するようにプロンプトされます。 詳細に関しては、以下を参照してください：データベースオブジェクトの追加、編集、および削除。

7.2.3.3 例："altova.mdb" データベースをマッピングに追加する

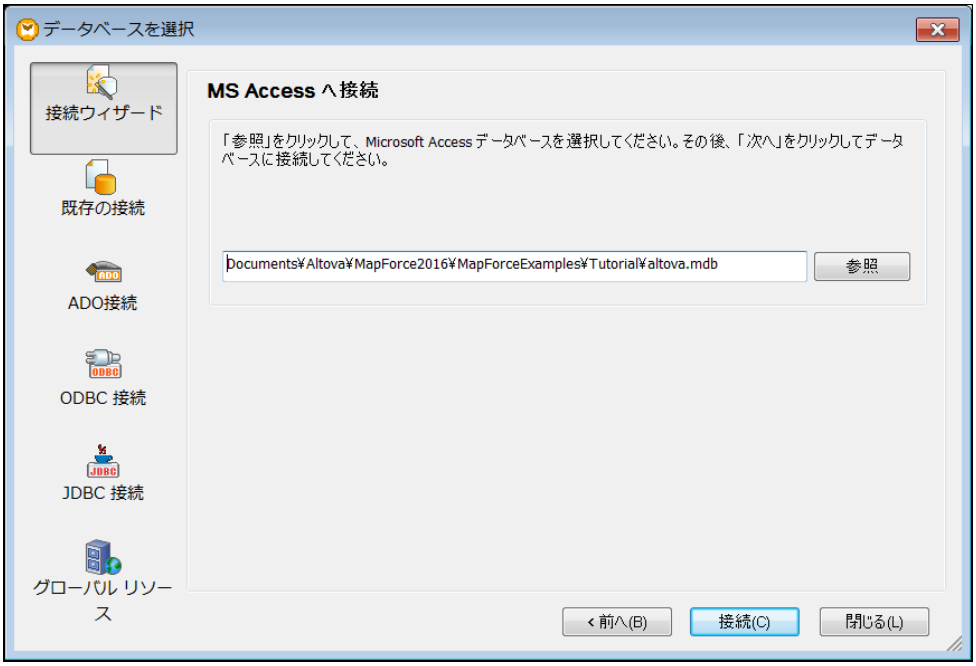
この例は、Microsoft Access データベースをマッピングに追加する方法を説明します。サンプルのデータベースは、**altova.mdb** と呼ばれ、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。**altova.mdb** データベースは、このドキュメント内で説明されているデータベースに関連したアクションと概念をサポートします。

altova.mdb データベースをマッピングに追加する：

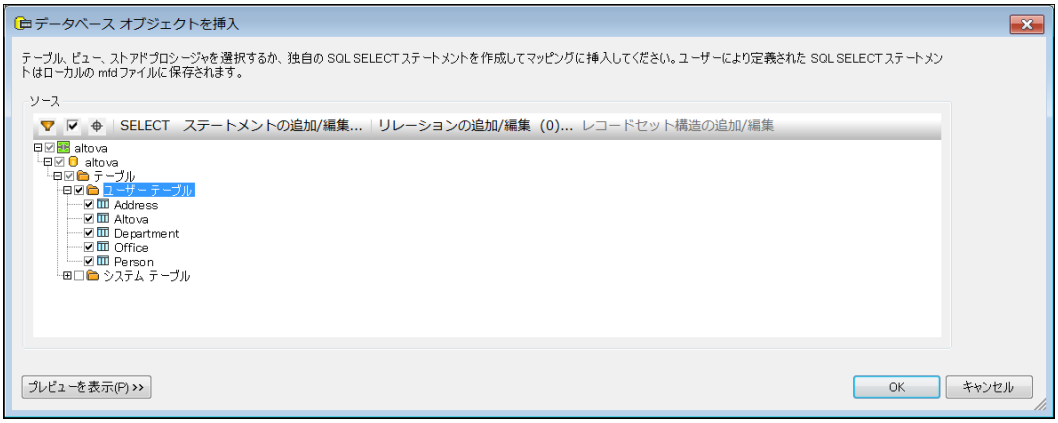
1. **「挿入」** メニューから **「データベース」** をクリックします。または、 **「データベースの挿入」**（ ）ツールボタンをクリックします。



2. **Microsoft Access (ADO)** をクリックして、 **「次へ」** をクリックします。



3. **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の**altova.mdb** ファイルを参照し、**接続**をクリックします。
4. データベースオブジェクトを選択するようにプロンプトされると、**ユーザーテーブル**を選択します。

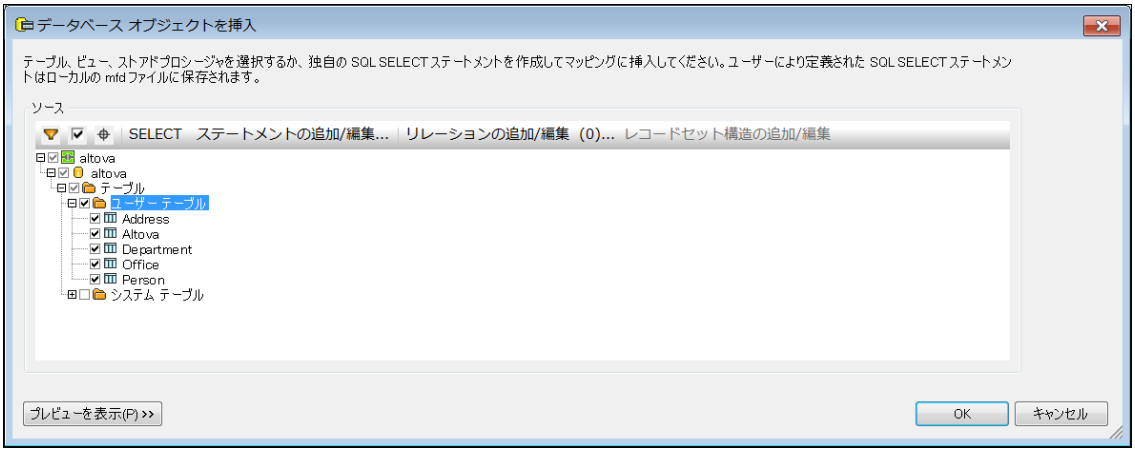


7.2.3.4 データベースオブジェクトを追加、編集、削除する

データベースの一部は、(スキーマ、テーブル、ビューなどの)多数のオブジェクトをもつことができます。このトピックは、マッピングの目的に必要なデータベースオブジェクトのみをマッピングする方法について説明します。下では、サンプルAccess データベースが使用されています。データベース型のための手順と類似しています。

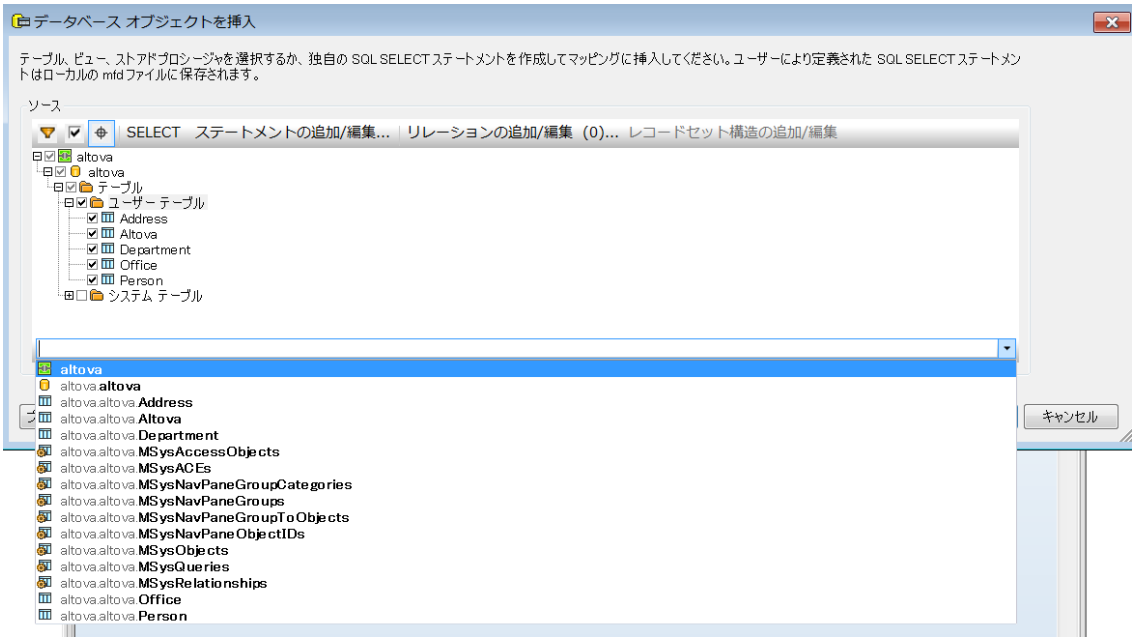
1. **挿入**」メニューから、**データベース**」をクリックします。
2. **接続ウィザード**」をクリックして、**Microsoft Access (ADO)** をクリックします。
3. **次へ**」をクリックし、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の**altova.mdb** を参照します。

マッピングに含まれるデータベースオブジェクトを選択することのできるアクションダイアログボックスが表示されます。

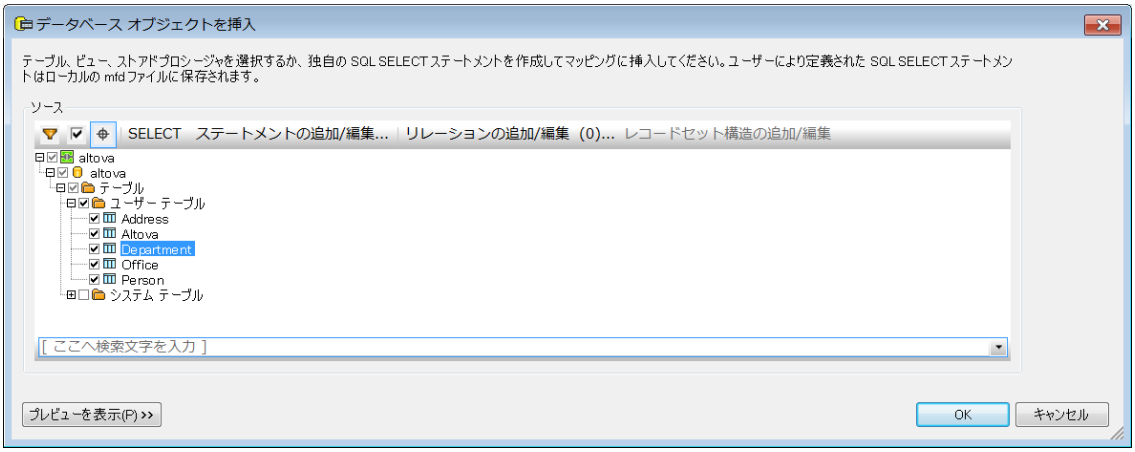


データベースオブジェクトを含むには、(例えば、テーブル)マッピングの横のチェックボックスをクリックしてください。この例の目的のために、**「ユーザーテーブル」**の横のチェックボックスをクリックします。

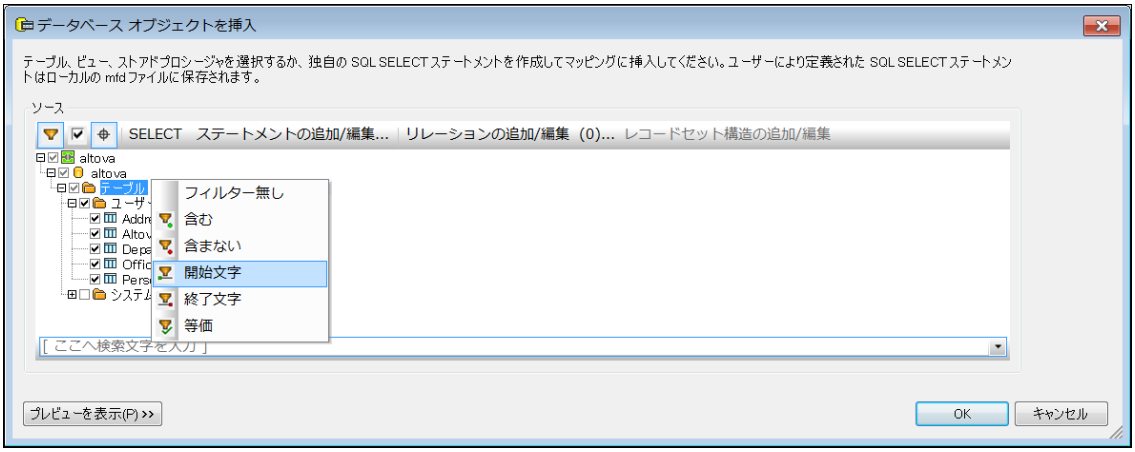
「オブジェクトロケーター」ボタン()を使用して特有のデータベースアイテムを検索することができます。ダイアログボックスの下の部分に表示される、コンボボックス内から特定のオブジェクトを選択(または、名前を入力)します。



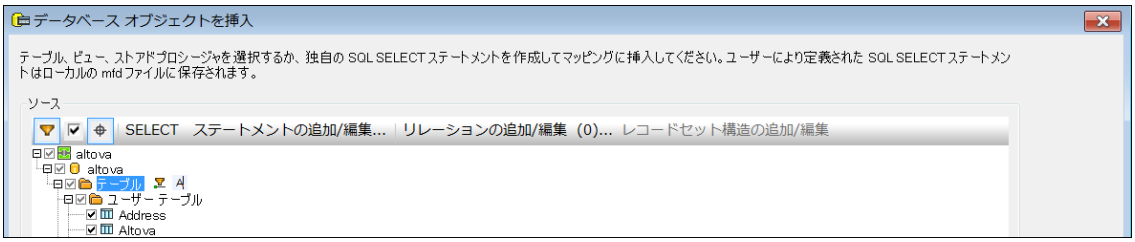
「フィルター」ボタン()によりオブジェクトを名前別にフィルターすることができます。1度フィルターボタンをクリックすると、フィルターアイコンがフィルタリングをサポートするオブジェクトの横に表示されます(この例では、"Tables")。



フィルターアイコンをクリックして、オブジェクト名の開始文字、終了文字、等価、または検索テキストを含むかなどを選択します。



フィルターの横に検索テキストを入力します (この例では、 'A'):



「チェックされたオブジェクトのみ表示」ボタン ( ) は、チェックボックスがアクティブな箇所のアイテムを表示します。

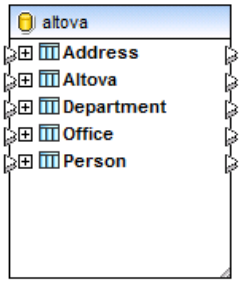
「SELECT ステートメントの追加/編集」ボタンを使用して、現在のデータベースのためにカスタム SELECT ステートメントを追加、または、編集することができます。このようなステートメントにより返されたデータは、この後マッピングソースとして使用することができます。 詳細に関しては、次を参照してください 仮想テーブルの SQL SELECT ステートメント。

「リレーションの追加/編集」ボタンを使用して、既存のものに加え、データベース内のフィールド間のローカルな主キー-外部キー リレーションシップを定義することができます。詳細に関しては、次を参照してください ローカルリレーションシップの定義。

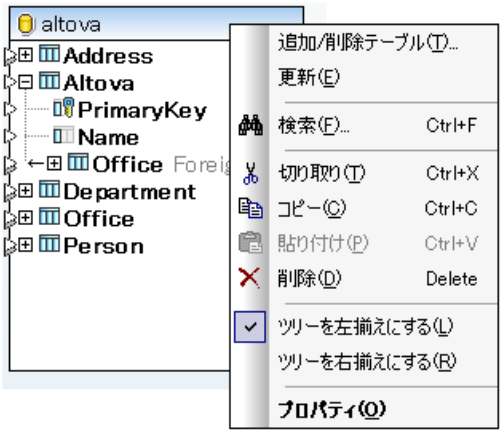
「レコードセット構造の追加 編集 」ボタンは、ストアドプロシージャをサポートするデータベースに適用されます。ストアドプロシージャがデータベースツリーから現在選択されている場合のみ有効化することができます。詳細に関しては、次を参照してください:ストアド プロシージャ。

「プレビューの表示」ボタンにより、現在選択されているテーブルから、または、ビューから素早くデータをプレビューすることができます。データベースブラウザーを使用して、マッピング処理とは別にデータベースを参照し、クエリすることもできます。詳細に関しては、次を参照してください:データベースの参照とクエリ

マッピングへデータベースオブジェクトを追加する準備が整うと、**「OK」**をクリックします。選択されたテーブル、ビュー、のみがデータベースコンポーネント上に表示され、マッピング接続をMapForce の標準の方法で描くことができます。



データベースオブジェクトを変更するには、コンポーネントを右クリックして、**「データベースオブジェクトの追加 削除 編集」**を選択します。



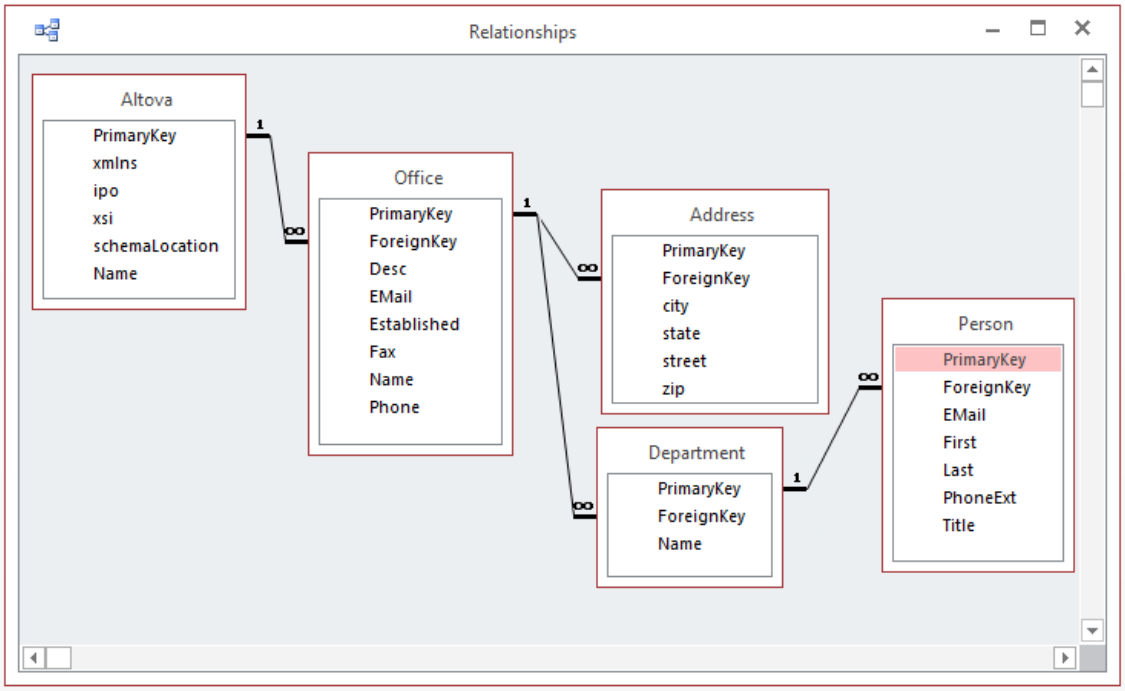
7.2.3.5 データベースリレーションシップの処理

リレーショナルデータベースは、その名前が示すように、テーブル間にリレーションシップを持ちます。フォルダー **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** にある**altova.mdb** データベースを例にしてみると、複数のリレーションシップが存在することに気がつきます:

- サンプルの会社（テーブル「Altova」に対応する）1つまたは複数のオフィスから構成されています（例えば、1つは東京に、もう1つは名古屋に存在するとします）。データベース用語では、「Altova」と「Office」テーブル間に1つ対複数のすなわち、「Altova」テーブル内の各**「PrimaryKey」**レコードのために、「Office」テーブル内で複数の**「ForeignKey」**レコードが存在することができます。**「ForeignKey」**値が、「Altova」内の

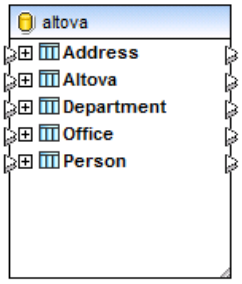
「PrimaryKey」値と対応する「Office」の箇所は、「Altova」のオフィスとして考えされます。

- 各オフィスは、1つまたは複数の部署（例えば、「Marketing」、「IT」、「Development」）から構成されています。「Office」と「Department」テーブル間に1つと複数のリレーションシップが存在します。
- 最後に、各部署は、複数の社員から構成されています。ですから、1つ対複数のリレーションシップは、「Department」と「Person」テーブル間のリレーションシップです。



altova.mdb データベース内のテーブル リレーションシップ（Microsoft Access "リレーションシップ"ビュー）

データベーステーブル間のリレーションシップは、マッピングの目的にとって重要です。MapForce は、データベースをマッピングに追加する際に、このようなデータベースリレーションシップの個々の変更を記録します。これにより、データをデータベースにマッピングする際、または、データベースにデータをマッピングする際、データベースリレーションシップを保持することができます。この概念をより良く理解するために、（**挿入 | データベース** メニューコマンドを使用して）**altova.mdb** データベースをマッピングに追加します。「ルート」テーブルの下のテーブルを呼び出します：



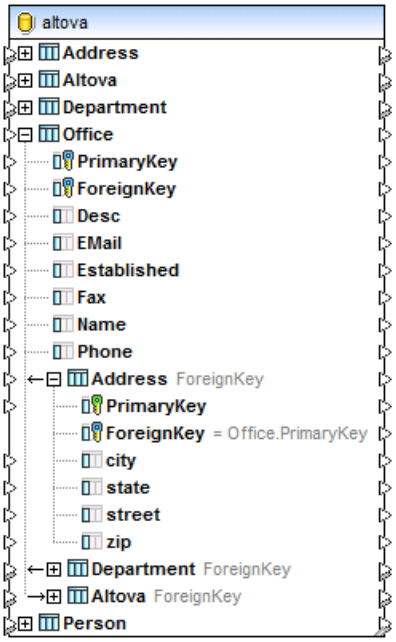
「ルート」テーブル

「ルート」テーブルを展開すると、ツリービュー内に存在する、すべての関連するテーブルが表示されます。例えば、**Office** テーブルを展開すると、関連したテーブルの階層が表示されます：

- テーブルの前の左矢印（←）は、このテーブルが子テーブルであることを表示しています。例えば、**Address** は、**Office** の子です。**Department** も**Office** の子であり、また、**Address** の兄弟テーブルであり、同じインデン

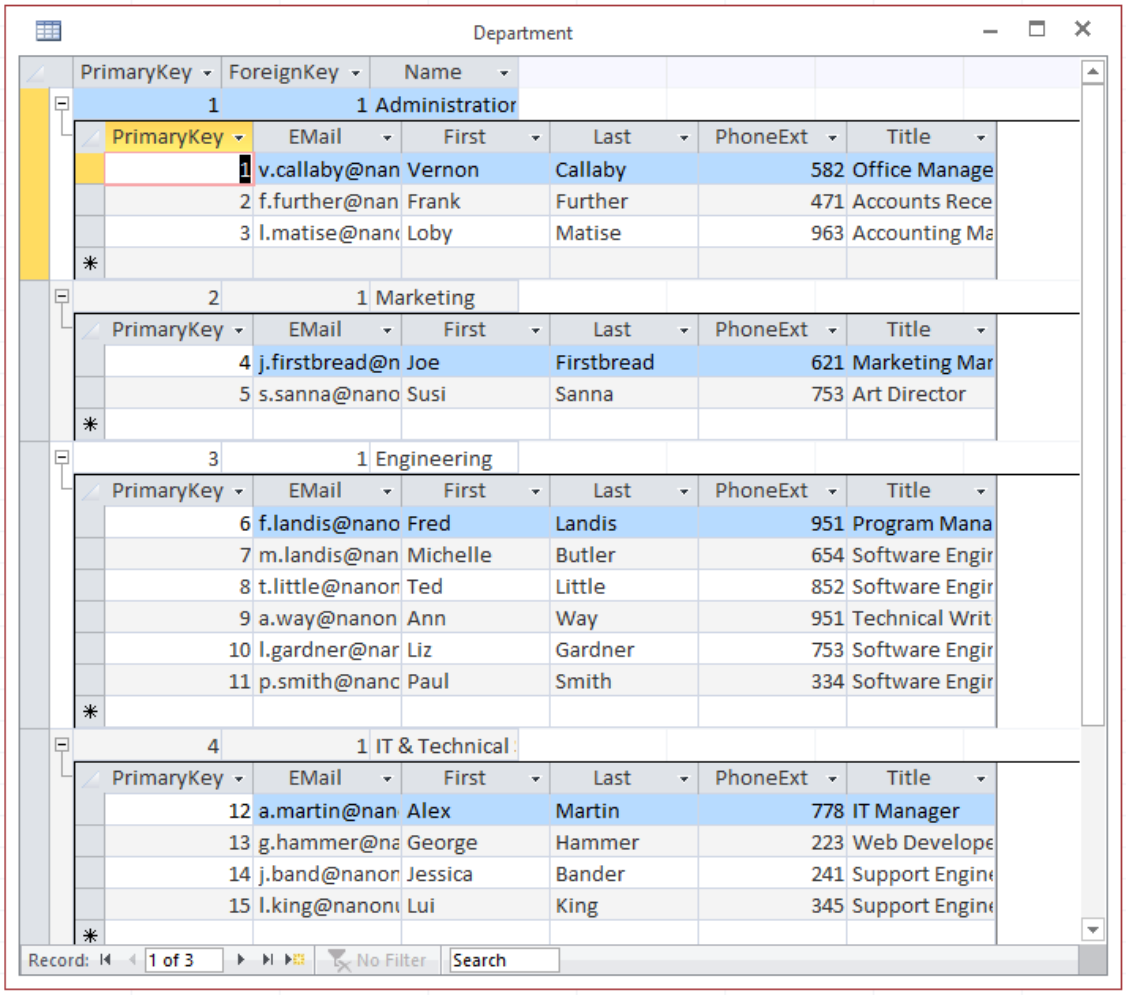
トのレベルに存在します。表示されているように、マッピング上の リレーションシップは、上の「リレーションシップ」ダイアグラムのに対応しています。

- テーブルの前の右矢印（→）は、このテーブルが親テーブルであることを表示しています。例えば、**Altova** は、**Office** の親です。



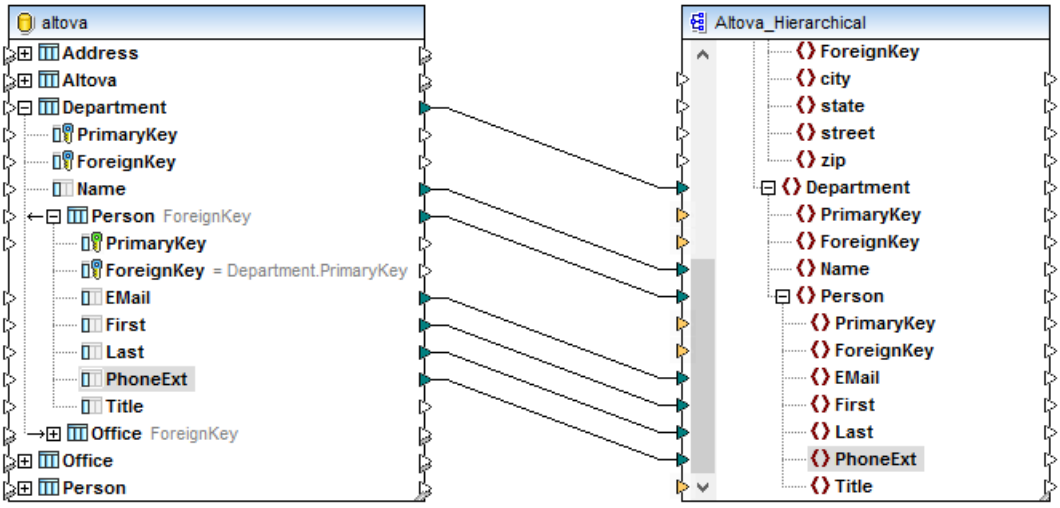
MapForce 内のテーブル リレーションシップ（altova.mdb データベース）

テーブルの階層の表示は、マッピングがデータベースから読み取る、またはデータベースへ書き込む際に既存のデータベースリレーションシップを保存する手助けをします。例えば、**Person** テーブルから全てのレコードを取得し、XML ファイルに各個人を部署にグループ別に入力するとします。具体的には、XML ファイルは、この例で使用されている**altova.mdb** データベースのように、各個人を部署にリンクします：



上記のように、「Administration」部署は、3人の社員、「Marketing」部署には、2人の社員、「Engineering」部署には6人の社員が所属しています。

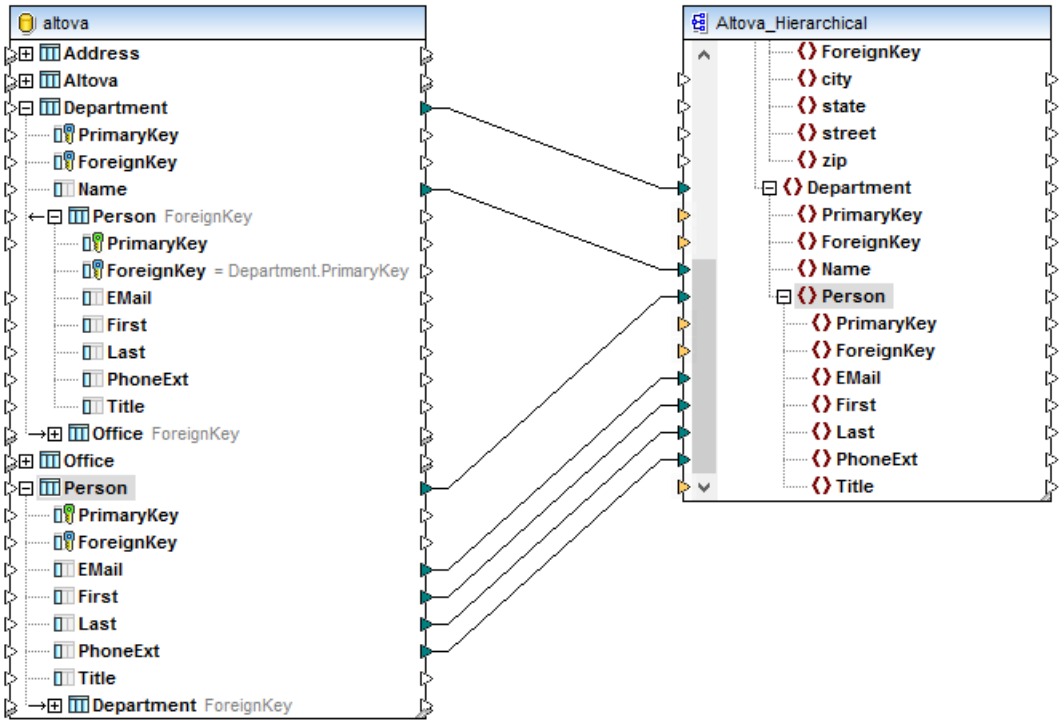
このデータベースからデータをマッピングする場合、各社員が正確な部署に割り当てられる場合、「ルート」テーブルとして、**Department** を使用することは重要で、**Department** の子である**Person** テーブルからマップします：



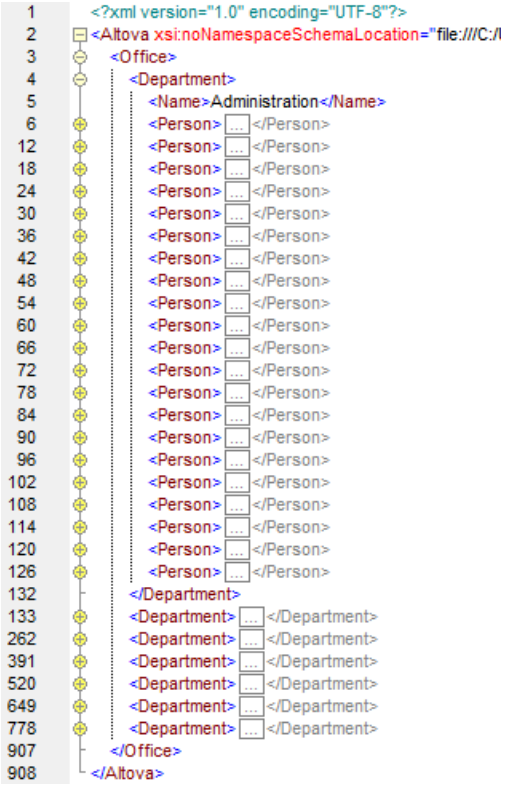
上のマッピングは、 **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の **DB_Altova_Hierarchical.mfd** ファイルに変更を加えたマッピングです。マッピングをプレビューする際には、各個人の結果は、意図とされる振る舞いであるとおり、部署別にグループ化されます。 「Administration」には、3人、「Marketing」には、2人、 「Engineering」には6人、などの結果が得られます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Altova xsi:noNamespaceSchemaLocation="file:///C:
  <Office>
    <Department>
      <Name>Administration</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Marketing</Name>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Engineering</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
  </Office>
</Altova>
```

若干修正の加えられた下のマッピングを確認してください。**Department** と**Person** の両方に接続が意図的に描かれているため両方が「ルート」テーブルとなっています。



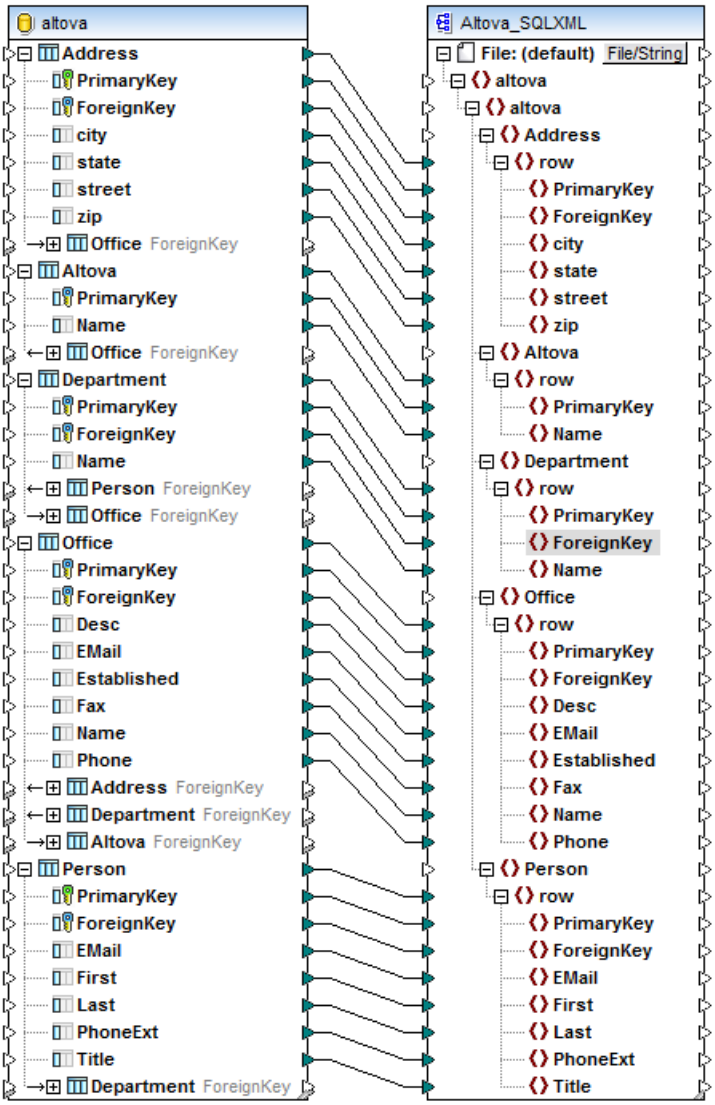
マッピングをプレビューすると、(ソースの部署に関係なく)すべての個人は、各ターゲット部署の下にグループ別に表示されています。これは意図とされた振る舞いではありません。つまり、つまり、「Administration」に21人の社員が存在し、「Marketing」に21人の社員が存在し、「Engineering」に21人の社員が存在するなどという結果を指します。



第 2番目の例では、データベースリレーションシップは、接続の方法のため無視されています。

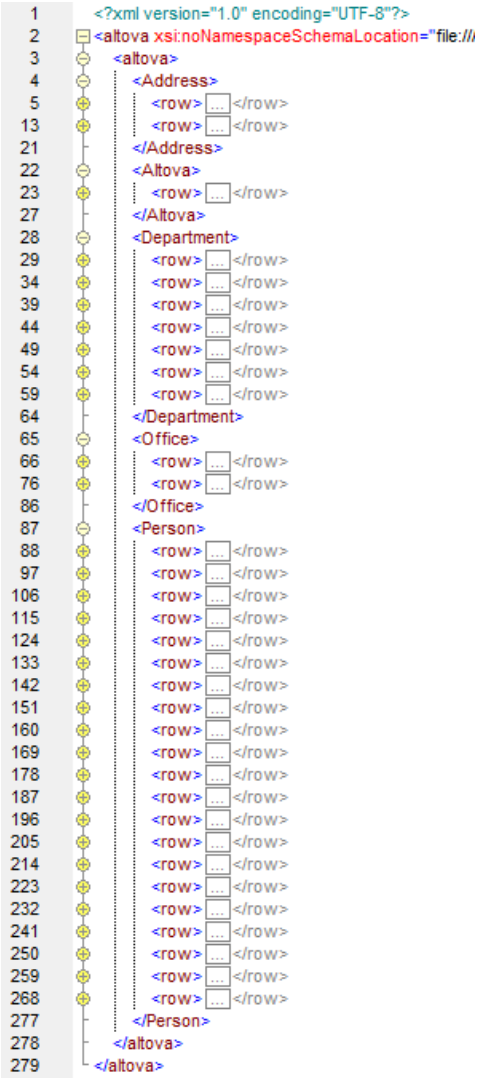
ですから、データベースリレーションシップを保持する場合、リレーションシップを保持する子テーブルを含むことを、または、同じ「ルート」テーブルから接続が描かれていることを確認してください。これは、ソースとターゲットデータベースと同じように作動します。リレーションシップを保持するデータベースマッピングの例に関しては: **<Documents>\Altova\MapForce2017 \MapForceExamples** フォルダー内にある**DB_Altova_Hierarchical.mfd** と **Altova_Hierarchical_DB.mfd** ファイルを参照してください。 次も参照してください: [データを複数にリンクされているテーブルに挿入する)。]

データベースリレーションシップを保持しない場合もあります。例えば、**altova.mdb** データベースからの全てのデータを、SQL/XML 仕様 (Structured Query Language (SQL) 仕様のパート 14に準ずる)フラットな XML ファイルにエクスポートすると想定します。この種類のマッピングは、 **<Documents>\Altova\MapForce2017 \MapForceExamples** フォルダ内の**DB_Altova_SQLXML.mfd** サンプルにより説明されています。マッピングの目的は、フラットな XML ファイルとしてデータベースデータを取得することです。ターゲット SQL/XML スキーマは、XMLSpyで **変換 | DB 構造からXML スキーマを作成する]** メニューコマンドを使用して生成することができます。



DB_Altova_SQLXML.mfd

上記のように、各データベーステーブルには、対応する要素がターゲットXML内に存在します。マッピングの結果をプレビューすると、実際のデータベースの行は、ターゲット内の「行」要素に書き込まれていることが確認されます。



XML 出力が示すように、XML 要素間には、階層は存在しません。これはフラットな SQL/XML 構造です。複数の「ルート」テーブルからのデータが内部でマップされたために、データベースリレーションシップは、無視されます。

7.2.3.6 ローカル リレーションシップの定義

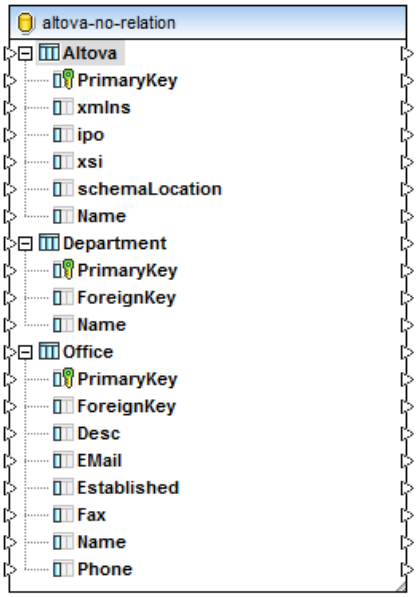
データベース テーブルが明示的にリレーションシップを定義していない場合、MapForce 内でそのようなリレーションシップをローカルに定義することができます。特に、MapForce 内から、異なるテーブルの列の間の主キーと外部キーリレーションシップをデータベースに影響を与えることなく作成することができます。すべてのデータベース列を主キー または 外部キーを作成するために使用することができます。また、データベース内に既存のリレーションに加え、新規のリレーションを素早く作成することができます。ローカルで定義されたリレーションシップはマッピングと共に保存されます。

これらの素早く作成されたリレーションシップは、MapForce 内では **ローカルリレーション** と呼ばれます。ローカルリレーションを次のデータベースオブジェクトのために定義することができます：

- データベーステーブル
- データベースビュー

- ストアドプロシージャと関数
- ユーザー定義 SELECT ステートメント

この例で使用されている、**altova-no-relation.mdb** データベースは、MapForce により与えられる**altova.mdb** データベースの簡素化されたバージョンです。Person ならびにAddress テーブルや、関連するテーブルがMicrosoft Access。**altova-no-relation** ツリーにあるテーブルは、全て『ルート』レベルのテーブルで、子テーブルはありません。各テーブルのコンテンツは、そのテーブルに含まれるものだけに限定されています。

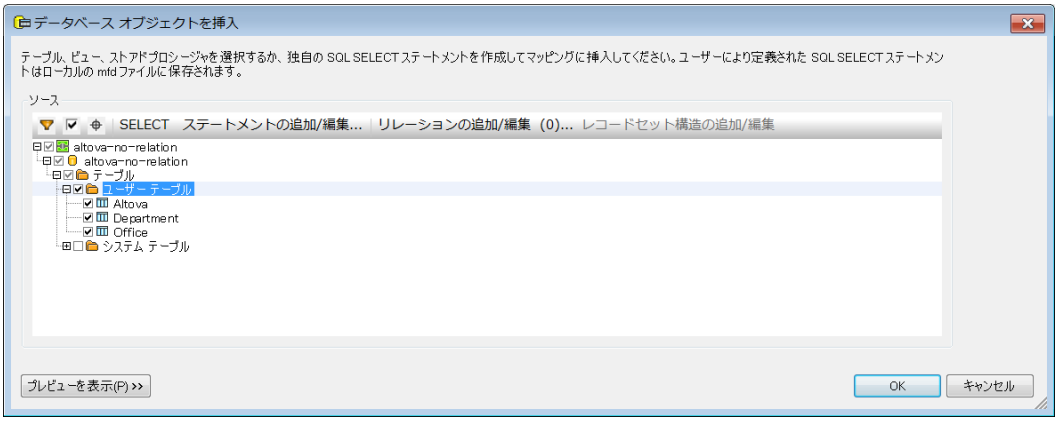


明示的なリレーションシップを持たないデータベース構造

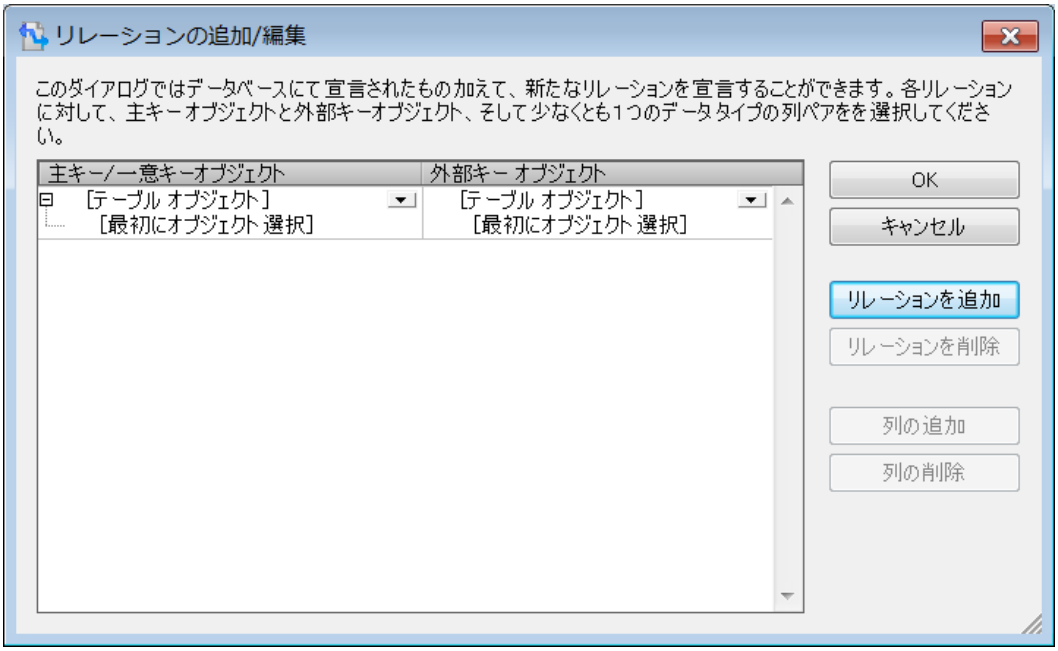
この例の目的は、「Altova」のオフィスを表示し、各オフィス内の部署を表示することです。**altova-no-relation.mdb** 内では、主キーと外部キーリレーションシップは、上記のように、明示的には存在しません。このリレーションシップは、論理的に (明示的に)存在し、マッピングの目的を完成するために ローカルで内で再作成されます。

ローカルリレーションは、データベースをマッピングに追加中に定義することができます、または、既存の データベース コンポーネントを右クリックして、または、コンテキストメニューから **「データベースオブジェクトの追加 削除 編集 」** を選択して定義することができます。

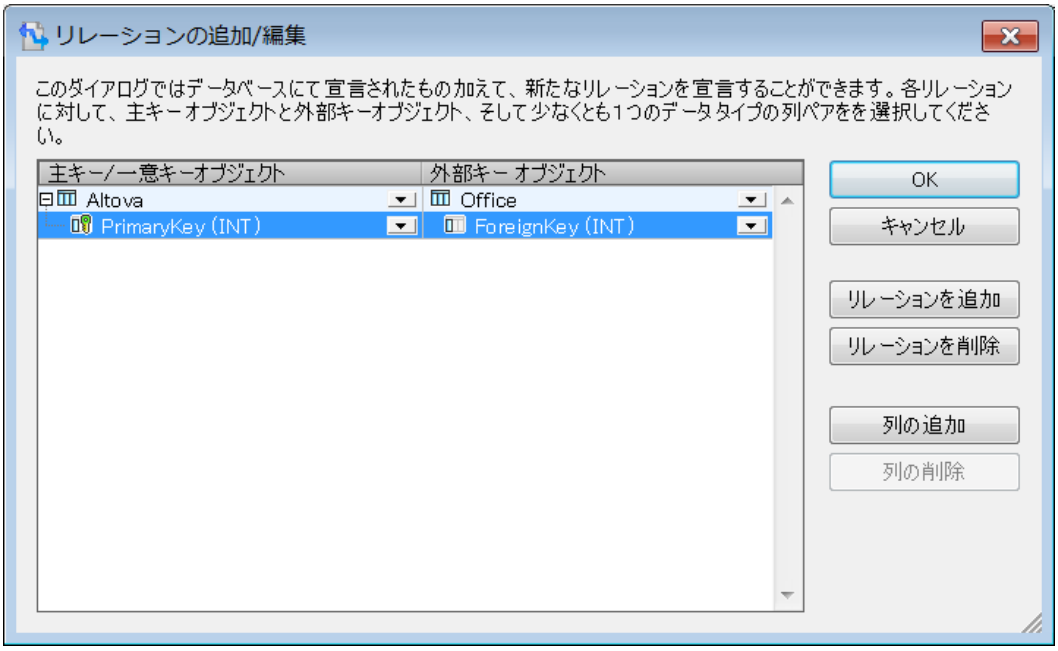
1. **「挿入」** メニューから、**「データベース」** をクリックします。
2. 「接続ウィザード」内の**Microsoft Access (ADO)** をクリックして、**「次へ」** をクリックします。
3. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にある **altova-no-relation.mdb** データベースを参照し、**「接続」** をクリックします。
4. **「ユーザーテーブル」** チェックボックスを選択します。



5. アイコンバーにあるリレーションを追加 編集ボタンをクリックします。
6. リレーションを追加 編集ダイアログボックスが開かれますので、**リレーションを追加ボタン**をクリックします。

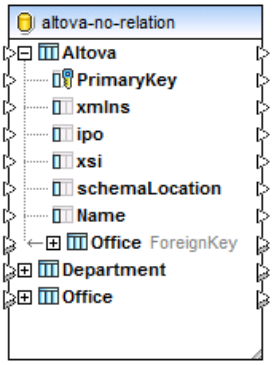


7. 2 つのコンボボックスによりテーブルまたはデータベースオブジェクト間のリレーションを選択することができます。左側のコンボボックスが主キー/ユニークキーのテーブルで、右側が外部キーのテーブルになります。主キー/一意キーオブジェクトがMapForce における親オブジェクトとなり、外部キーオブジェクトがデータベースコンポーネントの子として表示されます。(次も参照してください: データベースリレーションシップの処理)。

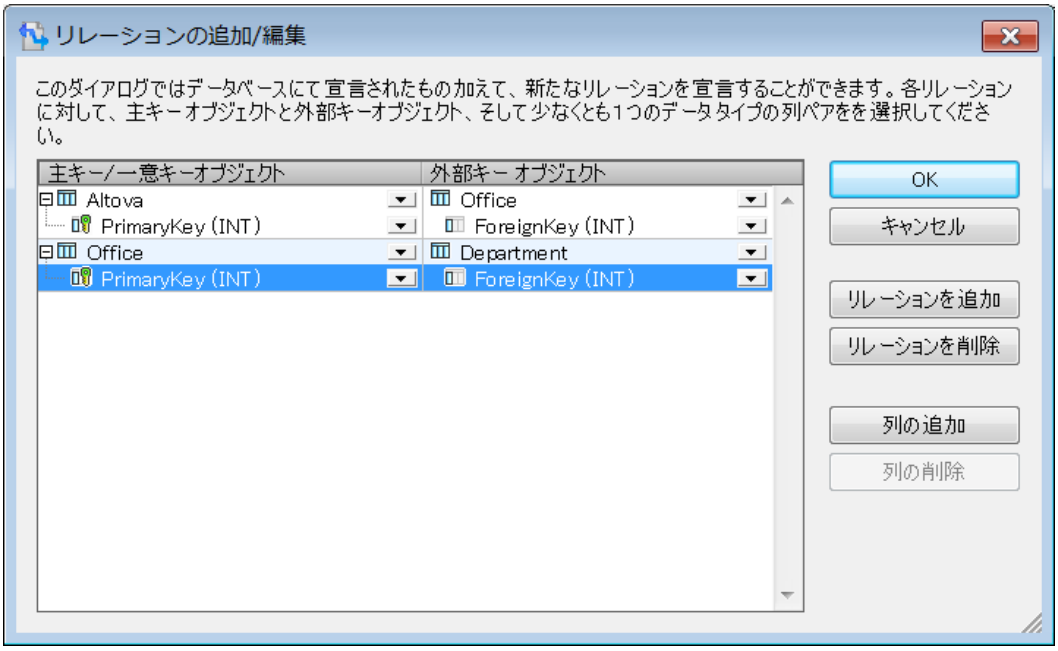


8. 「**OK**」ボタンをクリックしてローカルリレーションの定義を完了し、「**挿入**」ボタンをクリックして、データベースをマッピングエリアへ挿入します。

この段階では、「Altova」テーブルの「**PrimaryKey**」列と「Office」テーブルの「**ForeignKey**」列の間にローカルリレーションシップを作成します。下に表示されるとおり、「Altova」ルートテーブルは、「Office」テーブルの親になりました。具体的には、Office テーブルは、Altova テーブルの下の、自身の展開することのできるアイコンを持つ、関連するテーブルとして表示されています。



しかしながら、マッピングの目的はまだ達成されていません。マッピングの目的を完成するために、下に表示されるとおり、**Office** と**Department** テーブル間のリレーションシップを作成するメソッドと同じメソッドを使用します。

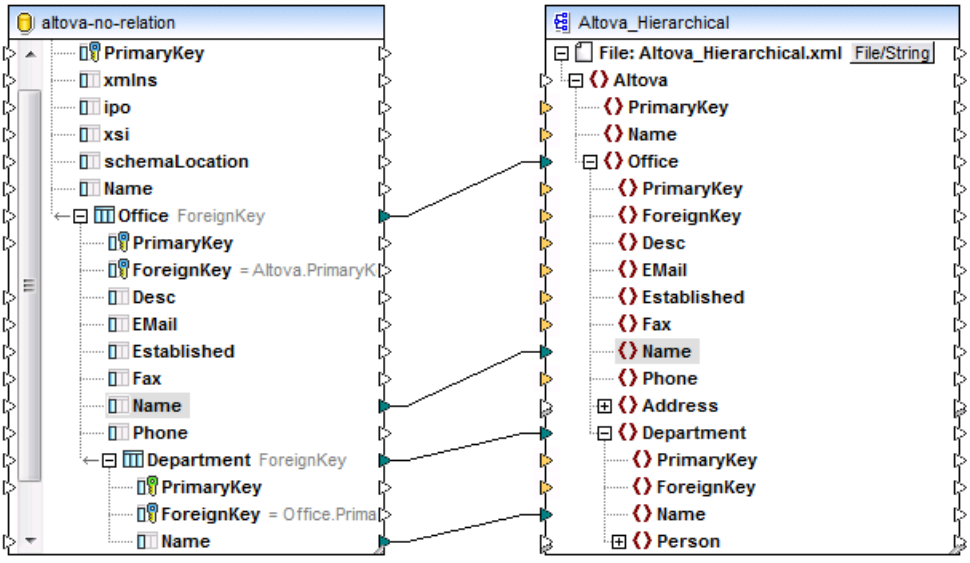


「リレーションの追加 編集」ダイアログボックスを再度開くには、データベースコンポーネントを右クリックし、 **データベースオブジェクトの追加 削除 編集...** コンテキストメニューから選択します。

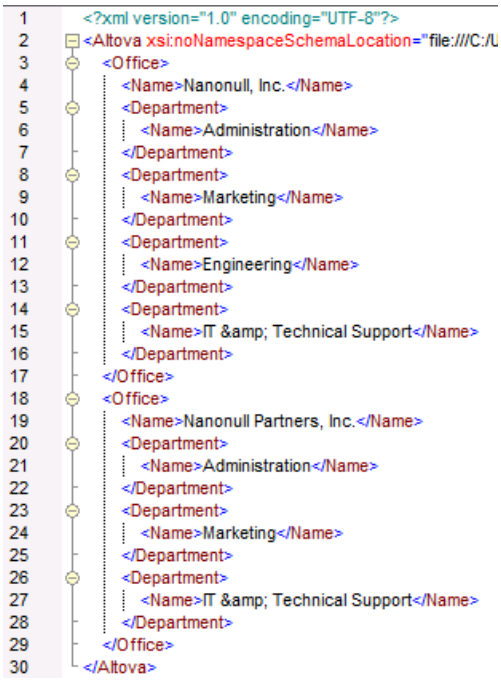
最後に、ターゲットスキーマをマッピングに以下のように追加します：

1. **挿入**メニューから、**XML スキーマ/ファイルの挿入...**をクリックします。
2. **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内で使用することのできる **Altova_Hierarchical.xsd** ファイルを参照します。サンプルXML ファイルを与えるようにプロンプトされると、**スキップ**をクリックします。ルート要素を選択するようにプロンプトされると、「Altova」を選択します。

ターゲットXML 内のテーブル間のリレーションシップを保持するために、全ての接続は、同じ「ルート」テーブルから、階層的(この場合、「Altova」)に描かれています。詳細に関しては、次を参照してください：データベースリレーションシップの処理。



マッピング上記のように、マッピングを定義し、 **出力」** タブをクリックして、結果をプレビューします。マッピングの結果は、このマッピングの目的のとおり、それぞれのオフィスの下にネストされた部署の要素を表示しています。



7.2.3.7 データベースを変更するマッピングの実行

(例えば、レコードの挿入、更新、削除など)マッピングがデータベースのデータを変更すると、変更は、マッピングを実行したエンジンによりデータベースに適用されます。マッピングを実行するエンジンは、MapForce、(スタンドアロンまたは FlowForce Server 管理下での)MapForce Server、または、C++、C#、またはJava のために生成されたコードの実行環境などです。

(**出力」** タブをクリックして、) MapForce 内で直接マッピングの結果をプレビューすると、更新スクリプトが表示されます。このスクリプトは、データベースに対して適用されたSQL ステートメントを表示します。スクリプトは、このアクションを明示的に実行するまで、データベースに対して実行されません。これはプレビューのためのみに使用することができます。

```
/*
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:
Data Source=C:/Users/altova/Documents/Altova/MapForce2016/MapForceExamples/AltovaTarget.mdb;Provider=Microsoft.Jet.OLEDB.4.0
*/

-- begin transaction

DELETE FROM [Address]

DELETE FROM [Person]

DELETE FROM [Department]

DELETE FROM [Office]

DELETE FROM [Altova]

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

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', '%PrimaryKey1%')
```

データベースを変更するマッピングの出力プレビュー (Altova_Hierarchical_DB.mfd)

上記の実行エンジン以外のSQL ツールをよう して、スクリプトをデータベースに手動で適用される必要があります。スクリプトは、外部 SQL エディターにより、理解されない値の書式を含む可能性があります。また、複数のアクションがテーブルに対して定義されている場合、(例えば、「以下の条件で更新し、残りを挿入」)第 2番目のアクションが条件付きで実行されるため、最初のアクションのみがプレビューに表示されます。

マッピングへの変更をデータベースへ直接 MapForce から適用する場合、**「出力」**メニュー内で使用することのできる**「SQL-スクリプトの実行」**コマンドをクリックします。このアクションは、実際にデータベースを変更し、効果はすぐに適用されます。

(スタンドアロンまたは FlowForce Server 管理下で)MapForce Server でマッピングが実行される場合、データベースへの変更はすぐに適用されます。生成されたコード内でも同様が発生します。コードをコンパイルし実行した際に、データベースへの変更は適用されます(例えば、Visual Studio 内で**「実行」**コマンドをクリックしてなど)。

> Your MapForce のインストールには、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内に存在する、複数のサンプルデータベース (Microsoft Access または SQLite ファイル)が含まれます。MapForce から**「SQL-スクリプトの実行」**コマンドを使用して、MapForce により与えられるサンプルの データベースに対してデータベースへの変更を適用することが奨励されません。これは、サンプルが使用できなくなる可能性があるからです。オリジナルのデータへの上書きを回避するために、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー全体のバックアップをファイルを更新する前に作成することが最善策です。

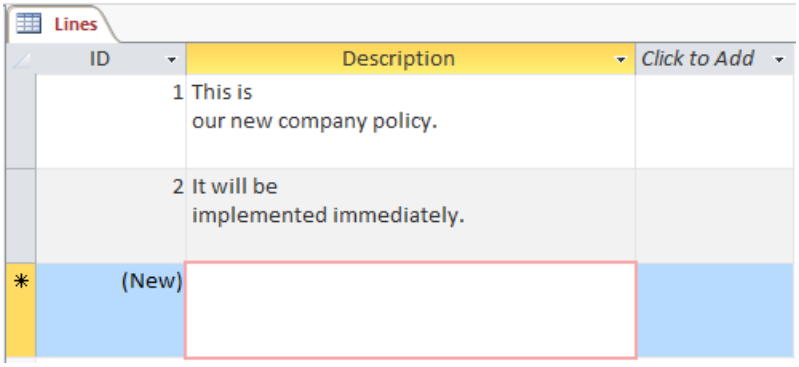
MapForce 以外の実行環境内でのマッピングの実行に関する情報は、以下を参照してください:

- FlowForce Server にマッピングをデプロイする
- MapForce Server 実行可能ファイルにマッピングをコンパイルする
- コードジェネレーター

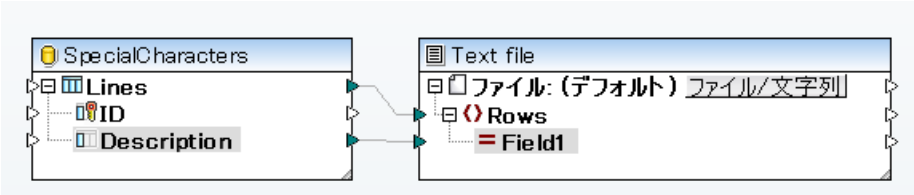
7.2.3.8 特殊文字の使用

データベースデータを変換する際に、データソースから、Enter キー改行 (CR/LF) 文字などの、固有の特別文字を削除する必要がある可能性があります。ソース。これは、MapForce ライブラリ関数**`char-from-code`** を使用して行うことができます。 .

テーブル「Lines」は、2つの列（「ID」と「Description」）から構成されています。Microsoft Access データベースが以下に表示されています。



目的は、CSV ファイルの詳細（各ラインごとの詳細）を抽出することです。ですから、この目的を達成するためのマッピングは以下に類似します：

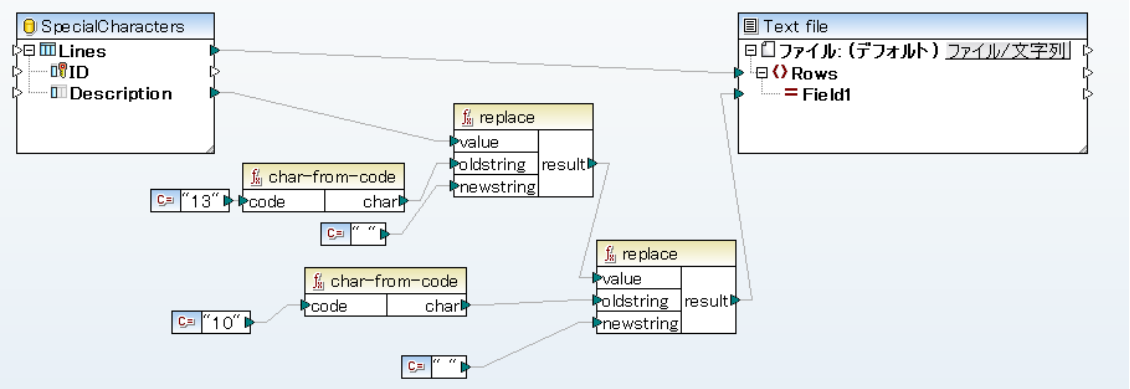


しかしながら、Access 内の各「Description」行は、意図する結果ではない、改行を含むCR/LF 文字により区切られた複数のラインを含んでいます：

```
1    "This is
2    our new company policy."
3    "It will be
4    implemented immediately."
5
```

この問題を解決するために、マッピングにMapForce ビルトイン ライブラリから**char-from-code** と**replace** 関数を追加します（次も参照してください：関数と作業）。上記で説明されている文字が発生すると、スペース文字で置き換えられるように、すべての詳細が処理されなくてはなりません。

Unicode チャート（http://www.unicode.org/charts/）では、LF とCR 文字は、**hex 0A | dec 10** と**hex 0D | dec 13** 文字にそれぞれ対応します。ですから、マッピングは、更に**replace** 関数を使用した処理を行うために、小数のUnicode 値 13を文字列に変換するように変更されなければなりません。



マッピングをこの時点でプレビューすると、各データベースフィールド内のCR/LF 文字 がスペースと置き換えられていることに注目してください。

```
1    This is  our new company policy.
2    It will be  implemented immediately.
3
```

7.2.3.9 Null 値の処理

マッピングランタイムでデータベースフィールドがnull 値であるかをチェックするには、**is-null** と**is-not-null** MapForce ライブラリ関数を使用します。MapForce から、テーブル内にnull フィールド が存在するかを確認するには、データベース ブラウザー を使用してクエリします (次を参照 : データベースの参照とクエリ)。

データベースフィールドをnull に設定するために、 **set-null** 関数を使用します。

データベースのnull 値を文字列と置き換えるには、 **substitute-null** 関数を使用します。この機能を説明する **DB_ApplicationList.mfd** は、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。

データベースを更新するマッピング内のNULL 値の比較に関しては、データベーステーブルアクション内のNull 値の処理を参照してください。

XML ドキュメントから、マッピングをデータベースにマップする、または、データベースからXML ドキュメントにデータをマッピングする際のnull 値の処理に関する情報は、次を参照してください Nil の値 /Nillable。

7.2.3.10 連続する、および、一意の値の生成

データを挿入したり、データベースを更新するには、マップされたソースからの入力がないデータベースフィールドのために シーケンシャル、または、一意の値を作成する必要がある場合があります。このような場合には、次のビルトインMapForce ライブラリ関数を使用します。:

- **auto-number** ("core | generator functions" ライブラリ内で使用することができます)。この関数は、数値フィールドのための主キーの値を生成するために通常使用されます。
- **create-guid** ("lang | generator functions" ライブラリ内で使用することができます)。この関数は、 特定のフィールドのためのグローバルに一意の識別子 (16進でエンコードされた文字列として) を作成するために使用されます。

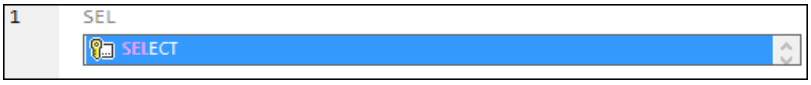
データベースにより生成された値を使用してデータベースフィールドのための値を書き込むこともできます。このオプションは、デ

ータベーステーブルアクションダイアログボックスで使用することができ、主キーを生成する際にとても役にたちます（次を参照：データベーステーブルアクション設定）。

7.2.3.11 SQL 自動補完の提案

特定のコンテキスト内にSQL ステートメントを入力すると、MapForce は、テキストのエントリを自動的に提案する場合があります。自動補完を次のコンテキストで使用することができます：

- SQL Editor (次を参照：データベースの参照とクエリ)
- 「データベーステーブルアクション」ダイアログボックス内の「カスタムSQL」テキストボックス（次を参照：データベーステーブルアクションの設定）
- 「SQL SELECT ステートメントの入力」ダイアログボックス（次を参照：SELECT ステートメントの作成）



自動補完

「上へ」と「下へ」キーボードキーを使用して、提案のリスト内を移動します。提案されたエントリを選択するには、クリックまたは「Enter」を押します。

自動補完の提案の無効化：

1. 「ツール」メニューから「オプション」をクリックします（または**Ctrl+Alt+O**を押します）。
2. 「データベース」から、「SQL エディター」をクリックします。
3. 「入力ヘルパー」から、「自動的に開く」チェックボックスをクリアします。

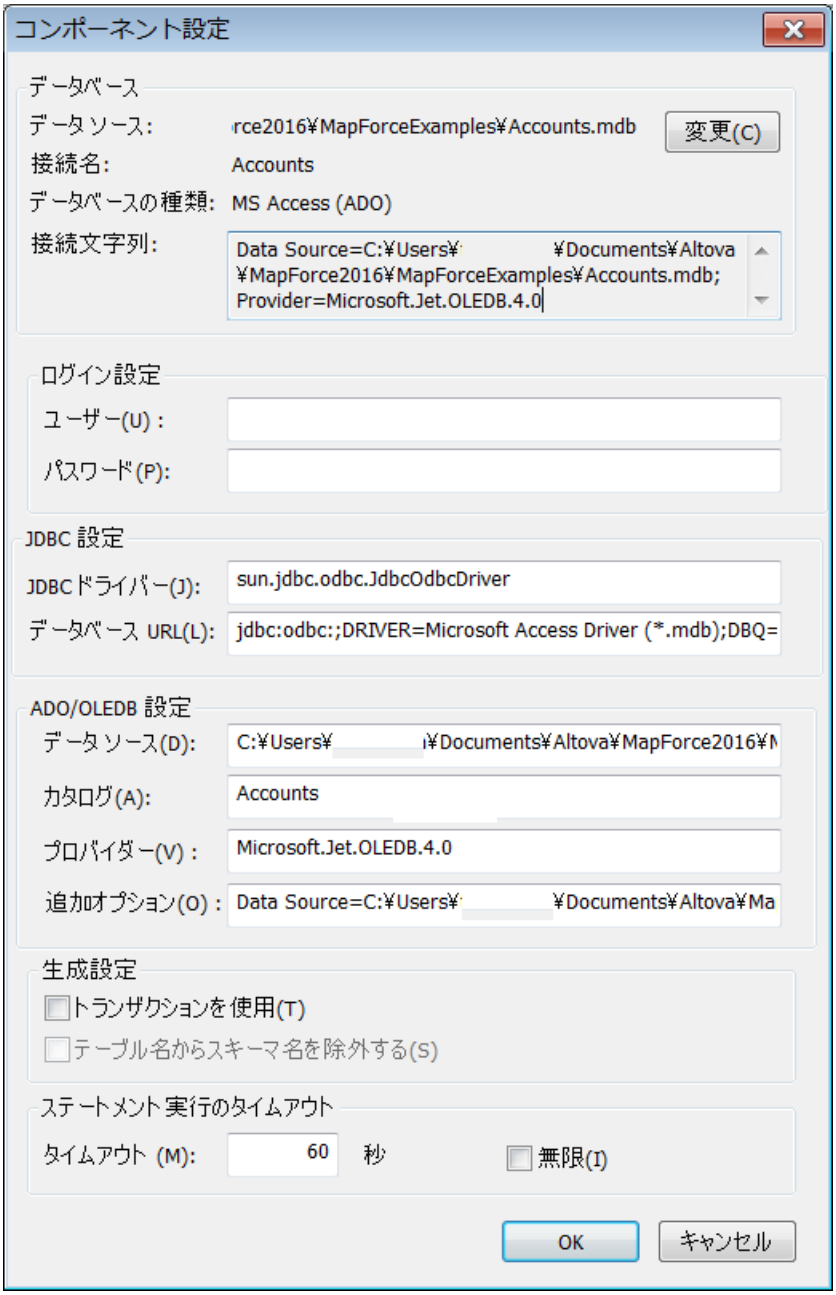
必要に応じて、自動補完の提案を手動で呼び出す：

- **Ctrl+Space** を押します。

7.2.3.12 データベースコンポーネントの設定

データベースコンポーネントをマッピングエリアに追加した後、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます：

- コンポーネントを選択して、「コンポーネント」メニューから「プロパティ」をクリックします。
- コンポーネントをダブルクリックします。
- コンポーネントを右クリックして、「プロパティ」をクリックします。



データベースコンポーネント設定ダイアログボックス

使用することのできる設定は、以下のとおりです。

データベース

このグループは、データベース接続情報を表示します。**「変更」**をクリックして異なるデータベースを選択し、既存のデータベースコンポーネント内のデータベースオブジェクトを再定義します。同じ名前のテーブルへの接続は保持されます。データベースコンポーネントを右クリックして、**「追加 削除 データベースオブジェクトの編集」**を選択することにより、コンポーネント内のテーブルを変更することができます。

データソース	現在のデータベースの名前を指定します。ファイルベースのデータベ

	ースでは、これはファイルシステム上のパスです。
接続名	接続の名前を指定します。 (データベースがデータソースを使用する場合、データソース名と同じです)
データベースの種類	データベースの型を指定します。
接続文字列	現在のデータベース 接続文字列を表示します。この読み取り専用のフィールドは、データベース接続の作成時または変更時に提供される情報を基に生成されます。

ログイン設定

ログイン設定はすべてのコード生成ターゲットとビルドイン実行エンジンのために使用されます。

ユーザー	データベースに接続するためのユーザー名を変更することができます。データベースが接続のためにユーザー名を必要とする場合必須です。
パスワード	データベースに接続するためのパスワードを変更することができます。 データベースが接続のためにパスワードを必要とする場合必須です。

JDBC-固有の設定

JDBC 固有の設定は Java コード生成のためのみに使用されます。

JDBC ドライバー	データベースコンポーネントのための現在アクティブなドライバーを表示します。デフォルトの ドライバー は、データベースコンポーネントを定義する際に自動的に入力されます。必要に応じてこの項目に入力されるドライバーを変更することができます。データベース URL フィールド内のエントリの構文が選択する特定のドライバに従っていることを確認してください。
データベースURL	現在選択されているデータベースの URL です。このエントリは JDBC-ドライバーフィールド内に入力されている特定の JDBC ドライバー 構文にしたがっていることを確認してください。

ADO/OLEDB-固有の設定

ADO/OLEDB は C++ と C# コード生成のためのみに使用されます。データソースとカタログ設定は、ビルドイン実行エンジンで使用されます。

データソース	ADO データソースの名前を表示します。
カタログ	ADO カタログの名前を表示します。
プロバイダー	データベースコンポーネントのための現在アクティブなプロバイダーを表示します。
追加オプション	追加のデータベースオプションを表示します。

生成設定

すべてのコード生成ターゲットとビルドイン実行エンジンに生成の設定は適用することができます。

トランザクションの使用	データベースをターゲットとして使用する場合、トランザクション処理を有効化します。エラーが発生すると、ダイアログボックスが開か

	れ、続行のオプションを選択できます。トランザクション処理は、このオプションを選択すると、すべてのデータベースコンポーネントのテーブルのために有効化されます。
テーブル名からスキーマ名を削除する	柔軟性のためにテーブル名だけを残し、データベーススキーマ名を生成されたコードから削除することが可能になります。 このオプションはにより生成された、MapForce SQL Select ステートメントのみに対して動作することにご注意してください。ユーザー定義 SQL-ステートメントは、仮想テーブルを作成する際には、変更されません。

ステートメント実行のためのタイムアウト
データベースがターゲットコンポーネントとして使用される場合、実行タイムアウトがサーバーの使用状況、交通、長期的に実行されているトリガー、および他の要素により発生する場合があります。この設定により、データベースへの接続が閉じられる前のタイムアウトの時間を定義することができます。設定は、データベースデータをクエリする際、C#、Java、および C++ コード内で実行されます。

タイムアウト	実行エンジンが、データベースステートメントの実行を中断する前にデータベースのレスポンスを待つ期間を秒数で定義します。実行タイムアウトのデフォルト設定は 60 秒です。
タイムアウトなし	有効化されると、このオプションは実行エンジンにタイムアウトしないように命令します。

メモ: *ステートメントの実行のためのタイムアウト*は SQLite データベースに使用することができません。

7.2.4 データをデータベースにマップする

このセクションは、MapForce によりサポートされるマッピングソースからデータを転送する例と方法について説明しています (例えば、XML ファイル) をターゲットデータベースに転送するなど。以下のテーブルは、使用することのできるオプションのまとめをリストしています。

希望するアクション	このトピックを参照する
マッピングにより提供されたデータを基にして、ターゲットデータベーステーブルにデータを挿入します。	• データをテーブルに挿入する
主キーの値の作成方法を管理します。	• データをテーブルに挿入する • データを複数にリンクされているテーブルに挿入する
「準備中」SQL ステートメントをテーブルがマッピングにより変更される前に実行します (例えば、テーブル、または、カスタム SQL ステートメント内の全てのレコードを削除するなど。)。	• データを複数にリンクされているテーブルに挿入する • データベーステーブルアクション設定
外部キーによりリンクされているテーブル内の階層的なリレーションシップを保持します。	• データを複数にリンクされているテーブルに挿入する
条件付きでテーブルを更新します。	• テーブルの更新
条件に従い、レコードをデータベースにマージします (レコードを更新して、他のレコードを同じテーブルに挿入します)。	• 「以下の条件で更新し、残りを挿入」アクション • MERGE ステートメント

希望するアクション	このトピックを参照する
外部キー リレーションシップを使用して他のテーブルにリンクされているテーブルを更新する際に、データベースの整合性を保持します。	• 親テーブルを更新する際の子テーブルのためのオプション
同じテーブルに対して複数のアクションを定義します（例えば、条件が満たされるとレコードは削除され、それ以外の場合、新規のレコードが挿入されます）。	• 「以下の条件で削除 ..」アクション • 「以下の条件で無視 ..」アクション
マッピングの実行に失敗した場合、データの整合性を保持します。	• トランザクション ロールバックの使用
データベース テーブルに複数のレコードをバルク挿入します（複数のINSERT ステートメントを1つのクエリにまとめます）。	• バルク転送 (MapForce Server)

7.2.4.1 データをテーブルに挿入する

マッピングは、他のデータベースを含む MapForce によりサポートされるソース コンポーネントからデータベース テーブルにデータを挿入することができます。新規に挿入されたレコードの主キーがどのように作成されるかを柔軟的に構成することができます。例えば、主キーは、データベースにより生成されたマッピングから取得、または、データベース テーブル内の既存のキーの値に基づき計算することができます。

この例は、新規のレコードをXML ファイルから既存の データベース テーブルに挿入する方法を説明しています。主キーの生成の構成方法についても説明されています。この例は以下のファイルを使用します：

- **altova-cmpy.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンス ファイルを検証するために使用されるスキーマです。
- **altova.mdb** ？ 更新されるターゲット データベースです。

全てのファイルは **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプル データベース ファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください：データベースを変更するマッピングの実行。

マッピングの目的は、です。 **altova.mdb** データベースの「Altova」テーブル内に、新規のレコードとして **Altova-cmpy.xml** 内の会社を入力することです。ソース XML ファイルを開くと、「Microtech OrgChart」と呼ばれる会社が1つのみが含まれていることに気がつきます。ですから、マッピングは、という名前「Microtech OrgChart」の新規のレコードを「Altova」テーブルに追加しなければなりません。また、新規の主キーが生成される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1：ソース XML コンポーネントを挿入する

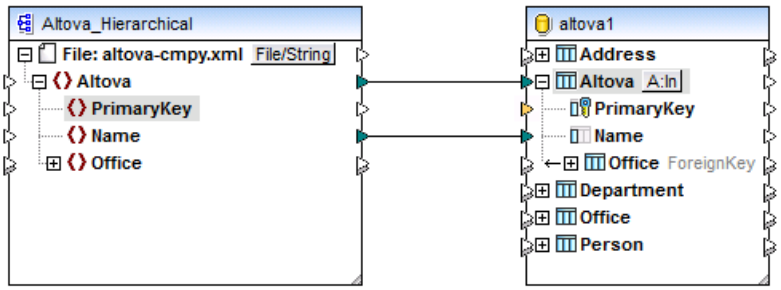
- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンス ファイルを与えるように問われると、**Altova-cmpy.xml** を参照してください。

ステップ 2：ターゲットデータベースを挿入する

- **「挿入」**メニューから、**「データベース」**をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。 次を参照してください：例："altova.mdb" データベースをマッピングに追加する)。

ステップ 3：接続を描く

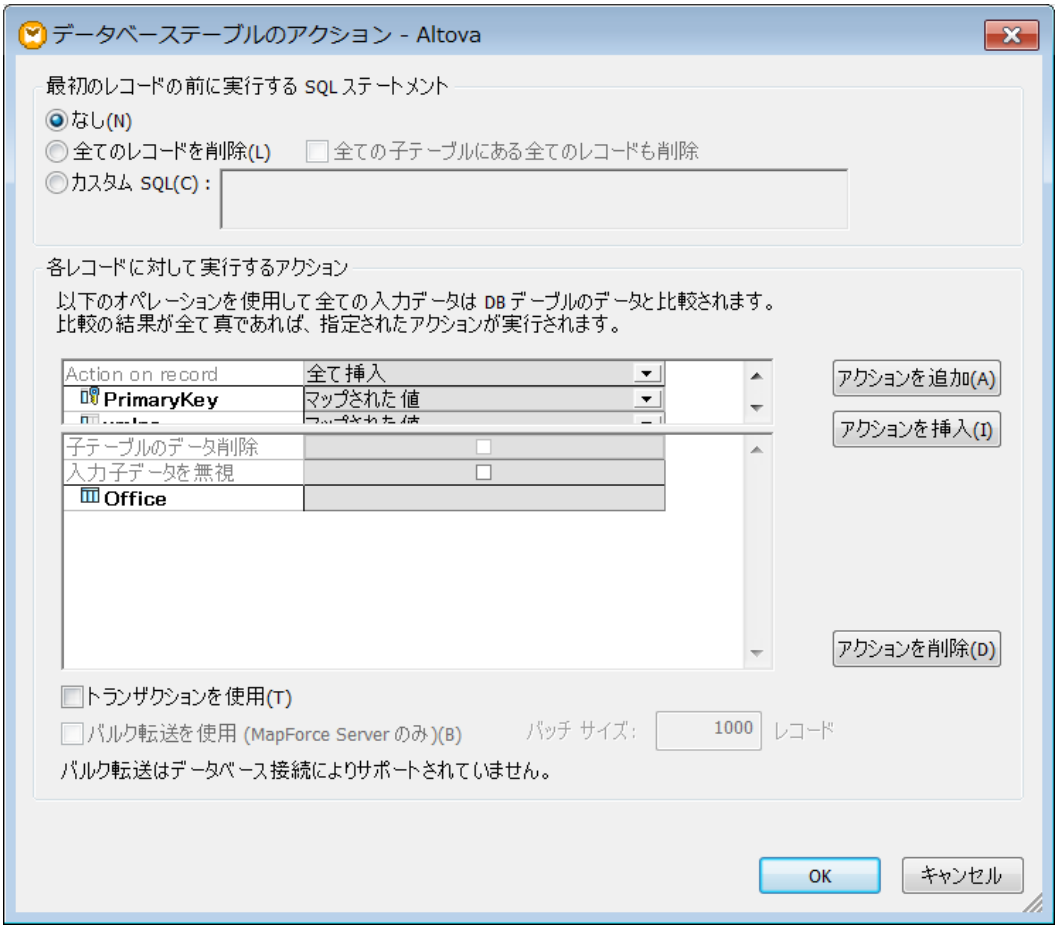
- 下に表示されるとおりマッピングの接続を描きます。



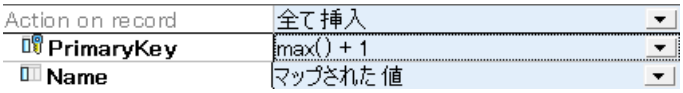
メモ： **希望しない接続が自動的に子孫アイテムに作成される場合は、「一致する子を自動的に接続する」**が有効化されています。この場合、最後のアクションを元に戻すには、メニューオプション **「編集 | 元に戻す」**を選択します。自動接続オプションを無効化するには、**「接続 | 一致する子を自動的に接続する」**を選択します。

ステップ 4：挿入アクションを構成する

1. ターゲットコンポーネント上に、**「アクション：挿入」**（ A:In ）ボタンが表示されていることに注意してください。このボタンは、マッピングからの接続がある各テーブルに表示されます（この場合は、「Altova」テーブル）。このボタンをクリックして、実行されるデータベースアクションの詳細を構成してください（この場合は、入力アクション）。データベーステーブルアクションダイアログボックスが表示されます。



2. データベーステーブルアクションダイアログボックス内で、 **全て挿入** から **「PrimaryKey」** の横の **max() + 1** オプションを選択します。



このリスト内で使用することのできるオプションは以下を意味します :

オプション	**説明**
マップされた値	ソースデータをデータベースフィールドに直接マップすることができ、全てのデータベースフィールドのための通常の設定です。リレーションを定義することにより、ストアドプロシージャを使用して、キーの値を与えることもできます。次を参照してください : ストアドプロシージャを使用した主キーの生成 .
max() + 1	データベース内の既存のキーをベースにして、キーの値を生成します。例えば、テーブルに主キー 1、2、および 3 を持つ 3 つのレコードが存在する場合、**max() + 1** は 4 になります。 この例では、「Altova」テーブルには、主キー 1 を持つレコードは 1 つしか存在せず、**max() + 1** は、期待される新規の主キーの値である 2 です。
DB-により生成された	データベースは、キーの値を生成するために **「Identity 関数」** を使用します。

「Name」の横のオプション**マップされた値**はこの列がマッピングから値を直接取得することを示しています。テーブルアクションダイアログボックスで使用することのできる他のオプションについての説明は、次を参照してください[データベーステーブルアクション設定](#)。

ステップ 5：マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために**出力**」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

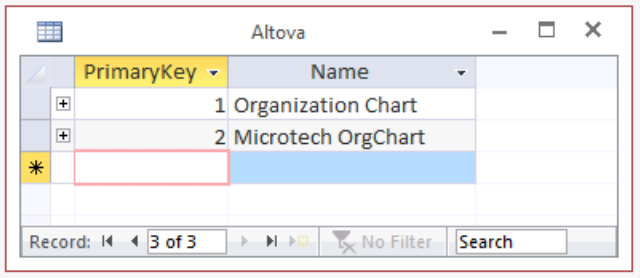
```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Insert\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
9   -->>> %PrimaryKey1%
10
11  INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
12
```

データベースに対してスクリプトを実行するには以下を行います：

- **出力**」メニューから、「**SQL-スクリプトの実行**」をクリックします。

メモ： MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください：[データベースを変更するマッピングの実行](#)。

結果を確認するには、DatabaseSpy または Access 内で**altova.mdb** データベースを開きます。新規の「Microtech OrgChart」レコードが「Altova」テーブルに新規の主キー 2 と共に、追加されていることに注意してください。このレコードためのデータは、入力 XML インスタンスに基づいたものです。



データベーステーブルにデータを挿入するマッピングの作成が完了しました。現在のテーブルと依存する子テーブルの両方にデータを挿入する例に関しては、次を参照してください[データを複数の関連するテーブルに挿入する](#)。

7.2.4.2 データを複数にリンクされているテーブルに挿入する

データベーステーブルは、「親」テーブルである可能性があります。つまり、このテーブルはデータベース内の他のテーブルに外部キーリレーションシップを使用して参照されている可能性があります。このような場合、マッピングに親テーブルだけでなく、依存する子テーブルにもレコードを挿入するように構成することができます。例えば、新規の"company" レコードをデータベーステーブルに挿入する場合、レコードをこの会社に関連したオフィス、オフィスの部署、社員などのためにも挿入することができます。

この例は、データベースリレーションシップを保持しつつ、複数のテーブルにデータを挿入する方法について説明しています。この例は前の例である[データをテーブルに挿入する](#)を若干高度にしたバージョンです：

- **Altova_Hierarchical.mfd** ？ 実際のマッピングファイルです。
- **Altova_Hierarchical.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **AltovaTarget.mdb** ？ 更新されるターゲットデータベースです。

全てのファイルは**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

マッピングの目的は、ソースからのデータXML ファイルを持つターゲットデータベース (**AltovaTarget.mdb**) 内のデータを置き換えることです。XML ファイル構成は、データベース内の テーブルの階層構造 に大体対応します。これは、以下のように構成されている組織図です：一番上の要素は、2つのオフィスと含む会社で、各オフィスには部署が存在し、各部署には社員が存在します。同じ階層は、「Altova」テーブルが会社に対応する**AltovaTarget.mdb** 内にも存在します。このテーブルは、外部キー リレーションシップを介して、「Office」テーブル内のレコードにリンクされています。同様に、「Office」は、「Department」にリンクされており、「Department」は、「Person」にリンクされています。**AltovaTarget.mdb** データベースのリレーションシップの図を確認するには、Access のリレーションシップビューでファイルを開きます (次も参照してください：データベースリレーションシップの扱い)。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1：ソース XML コンポーネントを挿入する

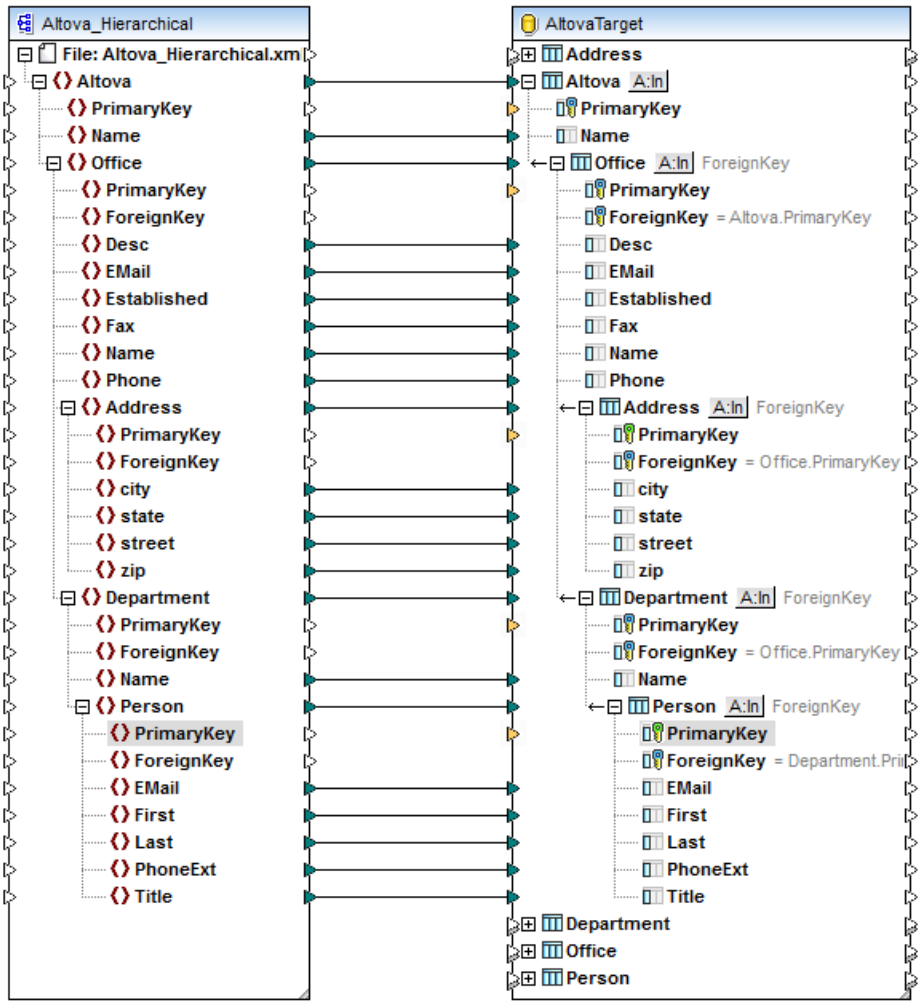
- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**Altova_Hierarchical.xml** を参照してください。

ステップ 2：ターゲットデータベースを挿入する

- **「挿入」**メニューから、**「データベース」**をクリックして、**AltovaTarget.mdb** に接続するためにウィザードのステップを行ってください。データベースへの接続方法は、altova.mdb への接続方法と同じです。 (次を参照してください：例："altova.mdb" データベースをマッピングに追加する)。

ステップ 3：接続を描く

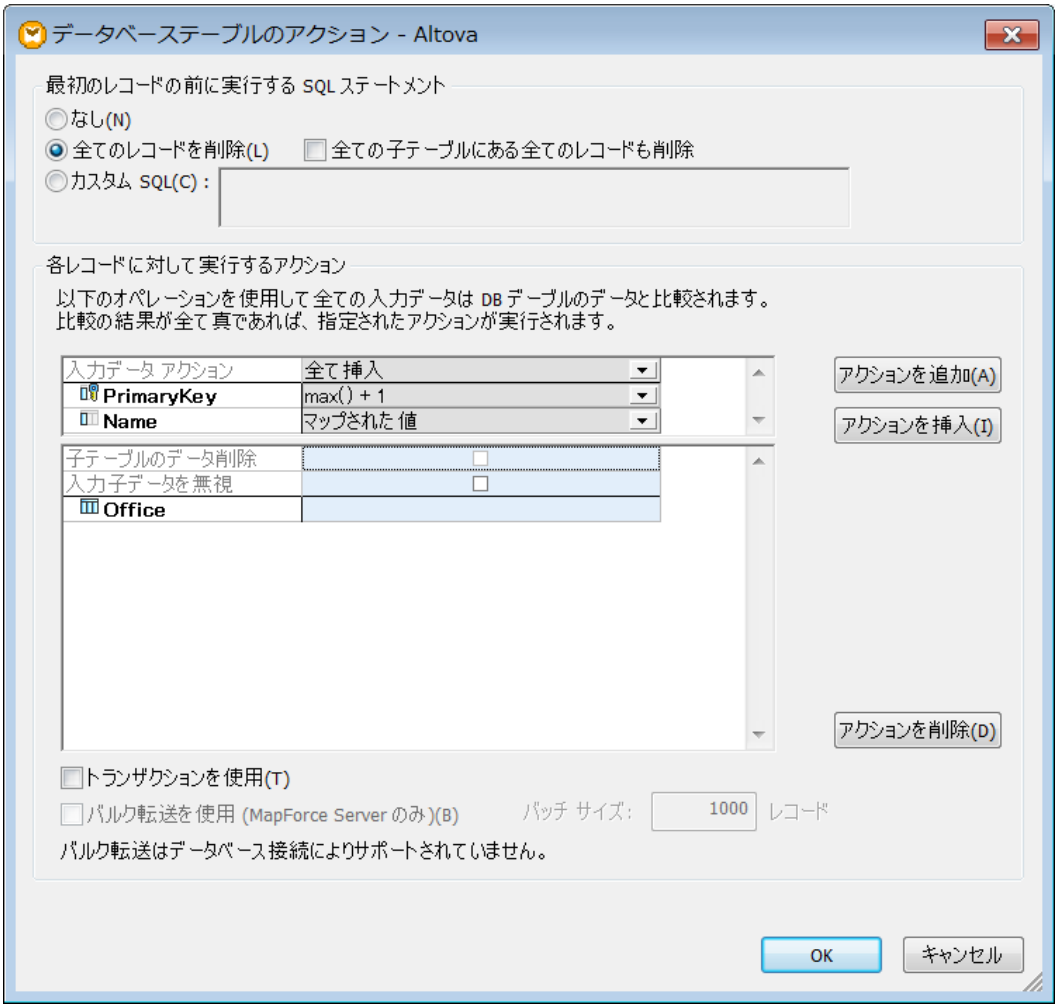
- 下に表示されるとおりマッピングの接続を描きます。主キーと外部キーがマップされていないことに注意してください。下に表示されるように素早く生成されます。



メモ: 希望しない接続が自動的に子孫アイテムに作成される場合は、「一致する子を自動的に接続する」が有効化されています。この場合、最後のアクションを元に戻すには、メニューオプション **編集 | 元に戻す」**を選択します。自動接続オプションを無効化するには、**接続 | 一致する子を自動的に接続する」**を選択します。

ステップ 4：挿入アクションを構成する

1. ターゲットコンポーネント上で、「Altova」テーブルの横の **アクション：挿入**（ A:In ）ボタンをクリックし、下に示されるとおり、主キーの **max() + 1** 設定を構成します。この設定に関しては、前の例で更に詳しく説明されています。次を参照してください：データをテーブルに挿入する。

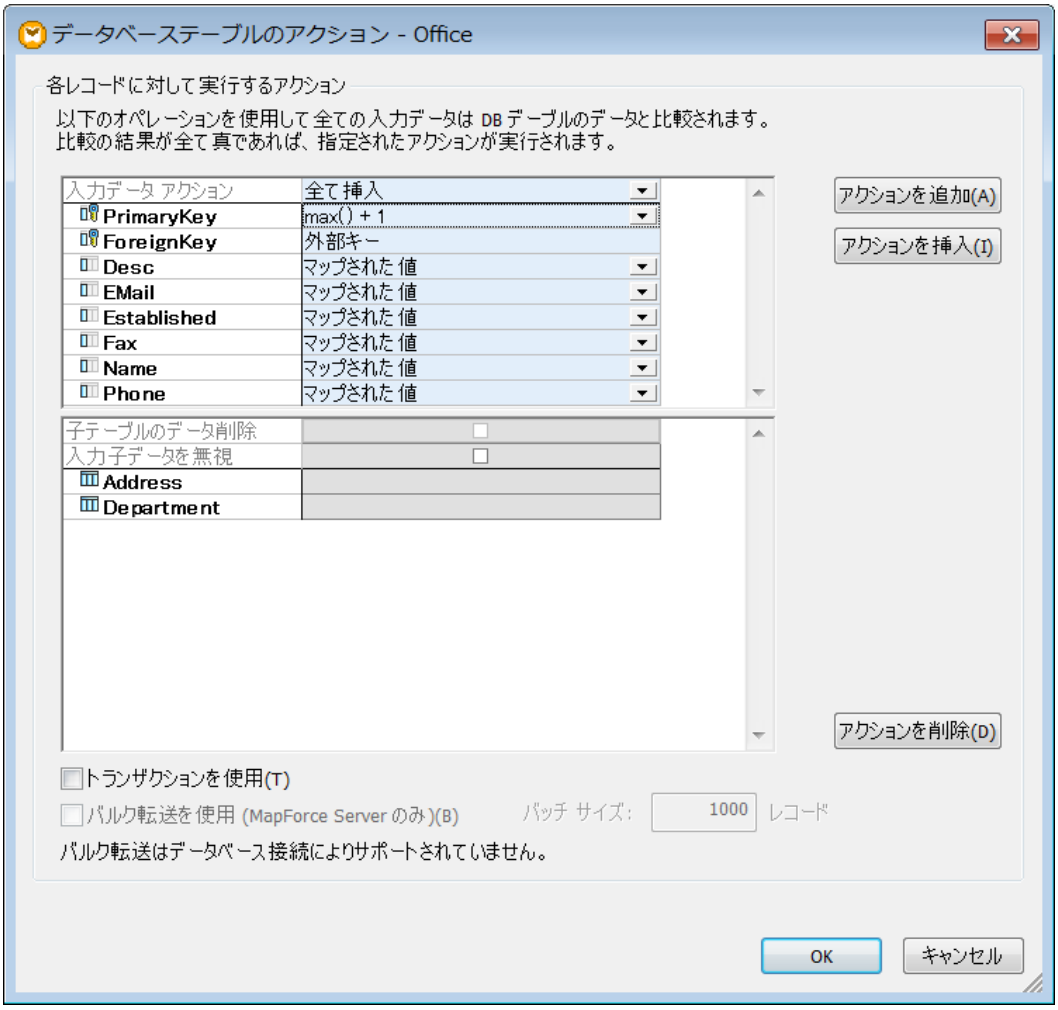


また、**「全てのレコードの削除」**オプションが有効化されていることに注意してください。これは、この例で希望される振る舞いどおり、新規のレコードが挿入される前に、テーブルからの既存のレコード全てをクリアします。このオプションを無効化すると、(新規の主キーを持つ)新規のレコードは、既存のレコードに加え、マッピングを実行する都度、既存のレコードに加え、データベースに追加されます。

この例のスコープのために、**「子テーブルから全てのレコードの削除」**オプションも有効化されています。このオプションにより、「Altova」テーブルからレコードが削除されるだけではなく、外部キー リレーションシップを使用して「Altova」テーブルにリンクされているすべてのテーブル内のレコードも削除します。子テーブルに子が存在する場合、はこれらの子テーブルも削除され、依存関係ツリー内の最後のテーブルまで削除されます。ルート「Altova」テーブルからレコードを削除することは、データベースの整合性の違反であり、マッピングの実行は、失敗します。

データベース テーブル アクション ダイアログボックスで使用することのできる他のオプションに関しての詳細は、次を参照してください: データベーステーブルアクション設定。

2. ダイアログボックスを閉じるために**「OK」**をクリックします。マッピング エリアで、ボタンの外観が DEL, A:In に変更されたことに注意してください。これは、"Delete" ステートメントが「挿入」アクションの前に実行されることが構成されていることを示します。
3. 「Office」テーブルの横の A:In ボタンをクリックし、主キーの**max() + 1** 設定を構成します。



4. 「Office」から降順し、各テーブルのためにステップ 3 を行います。具体的には、「Address」、「Department」、および「Person」です。これらすべてのテーブルがルート「Altova」テーブルの直接の子孫であることを確認してください。「ルート」テーブルの詳細とその重要性に関しては、次を参照してください:[データベースリレーションシップの扱い]。

ステップ 5 : マッピングをプレビューし、データベースを更新する

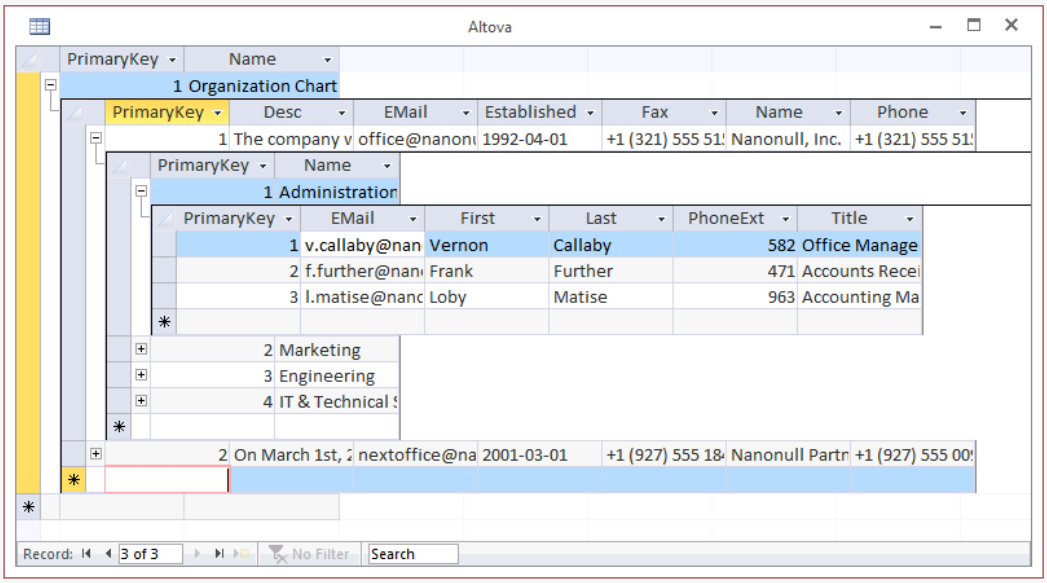
マッピングをプレビューするために **出力** タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

データベースに対してスクリプトを実行するには以下を行います :

- **出力** メニューから、**SQL-スクリプトの実行** をクリックします。

メモ: MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください:[データベースを変更するマッピングの実行]。

結果を確認するには、Microsoft Access で「Altova」テーブルを開き、「Altova」テーブルから「Person」テーブルに、XML ファイルからのリレーションシップがどのようにデータベースに伝達されているかを確認してください。



テーブルの整合性のリレーションシップを保持しつつ、複数の データベース テーブルへデータを挿入するマッピングの作成が完了しました。

7.2.4.3 テーブルの更新

この例は、XML ソースからデータが与えられる既存の データベース テーブルの更新方法について説明しています。この例は以下のファイルを使用します：

- **altova-cmpy.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** ？ 更新されるターゲットデータベースです。

全てのファイルは **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください データベースを変更するマッピングの実行。

マッピングの目的は、XML ドキュメントからの「Person」インスタンスを持つ「Person」テーブル内のレコードを更新することです。XML ファイル内の各 個人は、`PrimaryKey` 子要素を持ちます。「Person」テーブル内の各個人には、`PrimaryKey` 列が存在します。データベース内に `PrimaryKey` を持つ個人と XML ファイル内での個人の `PrimaryKey` が対応するレコードのみが更新される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1：ソース XML コンポーネントを挿入する

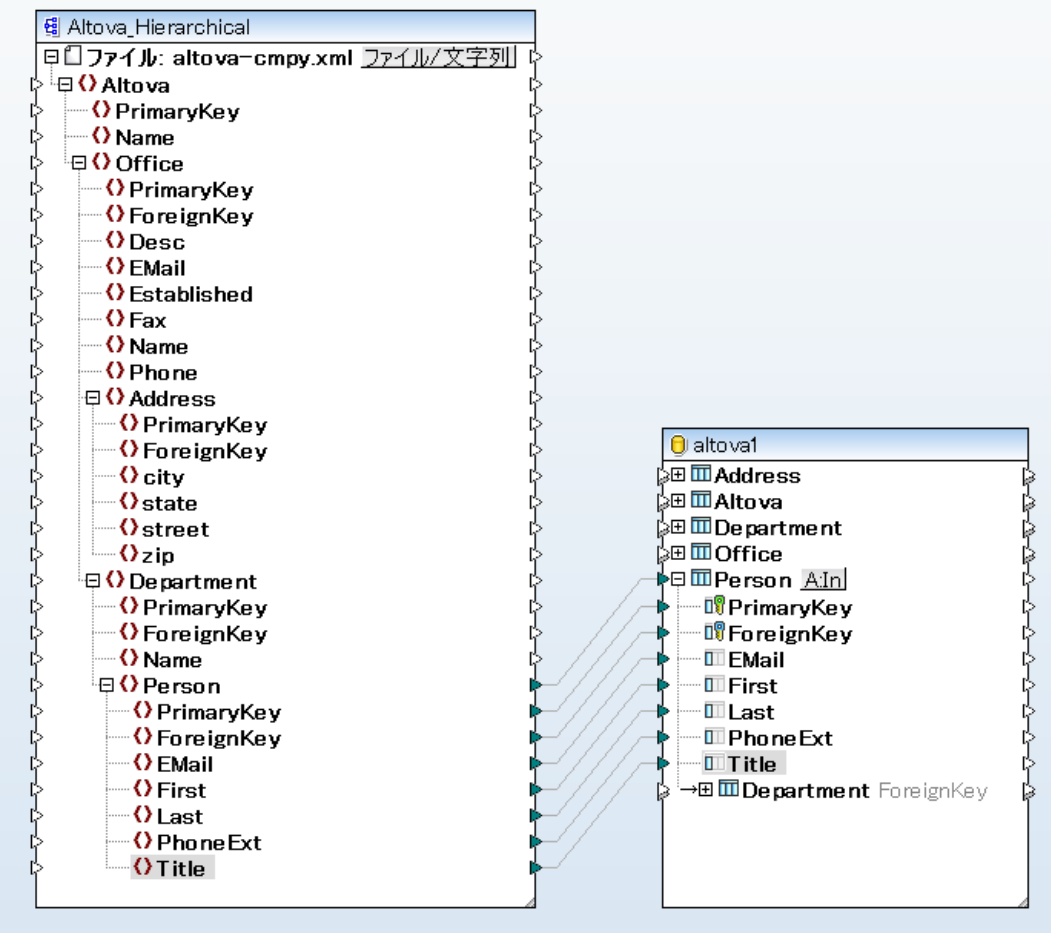
- **「挿入」** メニューから、**「XML スキーマ/ファイル」** をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy.xml** を参照してください。

ステップ 2：ターゲットデータベースを挿入する

- 「**挿入**」メニューから、「**データベース**」をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。次を参照してください：例：'altova.mdb" データベースをマッピングに追加する)。

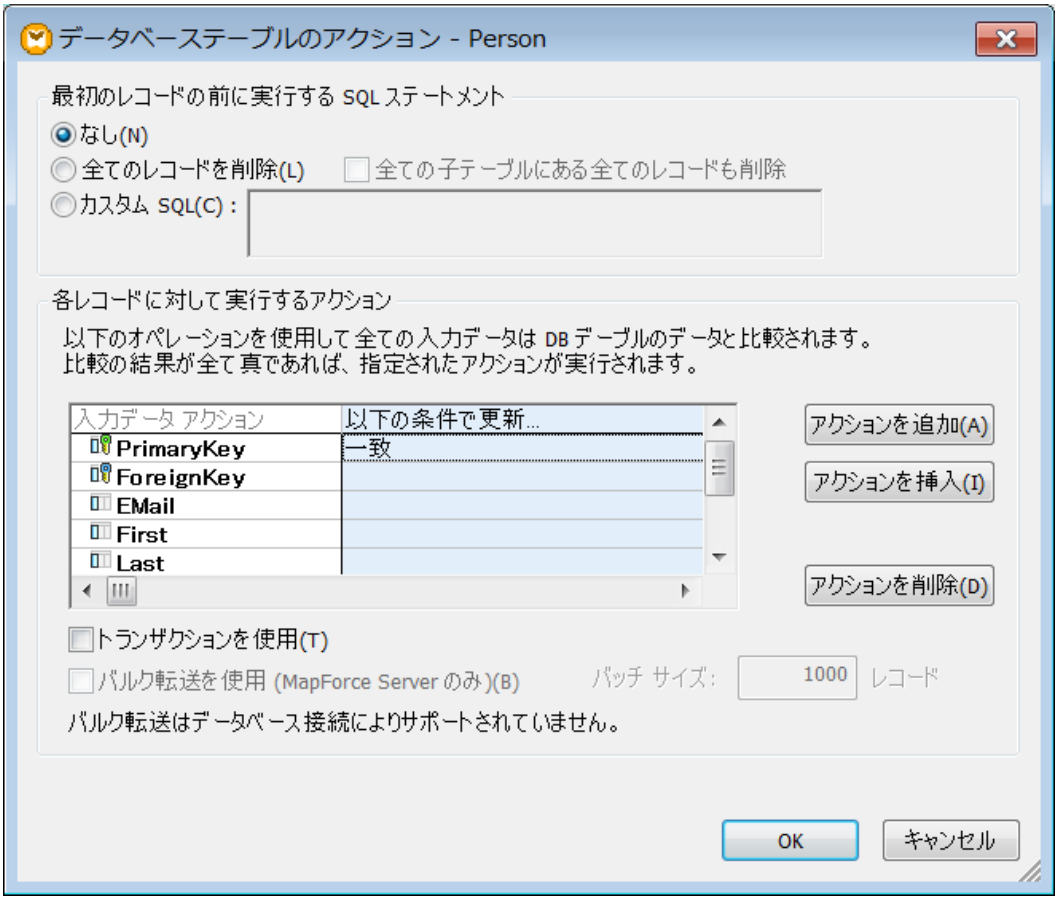
ステップ 3：接続を描く

- 下に表示されるとおりマッピングの接続を描きます。



ステップ 4：更新アクションの構成

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション：挿入**（A:In）ボタンをクリックします。
2. 「**レコードに対するアクション**」の横の「**以下の条件で更新する..**」、を選択します。これにより、データベーステーブルアクションを、条件付きの更新アクションに変更します。つまり、条件が満たされる場合のみ、現在のレコードは更新されます（次のステップを参照してください）。
3. 「**PrimaryKey**」値の横にあるオプションを、以下に表示されるように「**Equal**」に選択します。これは更新の条件を定義します。つまり、データベースの記録は、「**PrimaryKey**」値が、マッピングから来る「**PrimaryKey**」値に等しい場合のみ、更新されます。



この例では、等値演算子 が **「PrimaryKey」** フィールドに適用されています。これはデータベースを更新する場合、よくあるシナリオです。型のフィールドに対して定義することのできる定義は、必ずしも主キーではありません。例えば、**「最初」** と **「最後」** のフィールドの横のオプションに **「equal」** を選択することにより、ソースXML 内で、最初と最後の名前が同じであるレコードのみを更新します。

4. ダイアログボックスを閉じるために **「OK」** をクリックします。マッピングに戻り、**アクション:挿入** A:In ボタンが **アクション:更新** (A:Up) ボタンに変更されたことに注目してください。これはこのテーブルのために更新アクションが構成されていることを示しています。

ステップ 5 : マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために **「出力」** タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

```
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   The database was connected using the following connection-string:
6   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update
    if\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 row(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander',
    [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 row(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis',
    [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 row(s).
17
18  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] = 'Durnell',
    [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
19  -->>> OK. 1 row(s).
```

データベースに対してスクリプトを実行するには以下を行います：

- **「出力」**メニューから、**「SQL-スクリプトの実行」**をクリックします。

メモ： MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください：データベースを変更するマッピングの実行。

7.2.4.4 「以下の条件で更新し、残りを挿入」アクション

既存のレコードを更新するだけでなく、同じデータベーステーブルに新規のレコードを挿入する必要がある場合があります。このような場合には、MapForce では、「以下の条件で更新し、残りを挿入」アクションを使用することができ、これは以下のように動作します：

- **「以下の条件で更新する」**条件が true の場合、既存のデータベースの記録は、マッピングからのデータにより更新されます。
- **「以下の条件で更新する」**条件が false の場合、また、**「残りを挿入」**条件が存在する場合、新規のレコードが挿入されます。
- ソースファイル内に対応するエントリが存在しないデータベース内にレコードが存在する場合、これらのレコードは変更されません。

この例は、XML ソースからのデータをデータベーステーブルに更新と挿入して(マージする方法を説明しています。この例は以下のファイルを使用します：

- **altova-cmpy-extra.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** ？ 更新されるターゲットデータベースです。

全てのファイルは**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを使用することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください：データベー

スを変更するマッピングの実行。

マッピングの目的は、ターゲット「Person」テーブルにソースXML ドキュメントからのすべてのレコードをマージすることです。具体的には、ソースXML データ内の各レコードのために、マッピングは以下を行う必要があります：

- XML ファイル内の個人 の`PrimaryKey` が、データベース内の個人 の`PrimaryKey` に対応する場合、レコードは更新されます。
- 上記の条件を満たさないPerson テーブル内の既存のレコードは影響を受けません。
- XML ファイル内の個人 の`PrimaryKey` が、ターゲットデータベーステーブルに一致するものが無い場合、新規のレコードをデータベーステーブルに追加してください。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1：ソース XML コンポーネントを挿入する

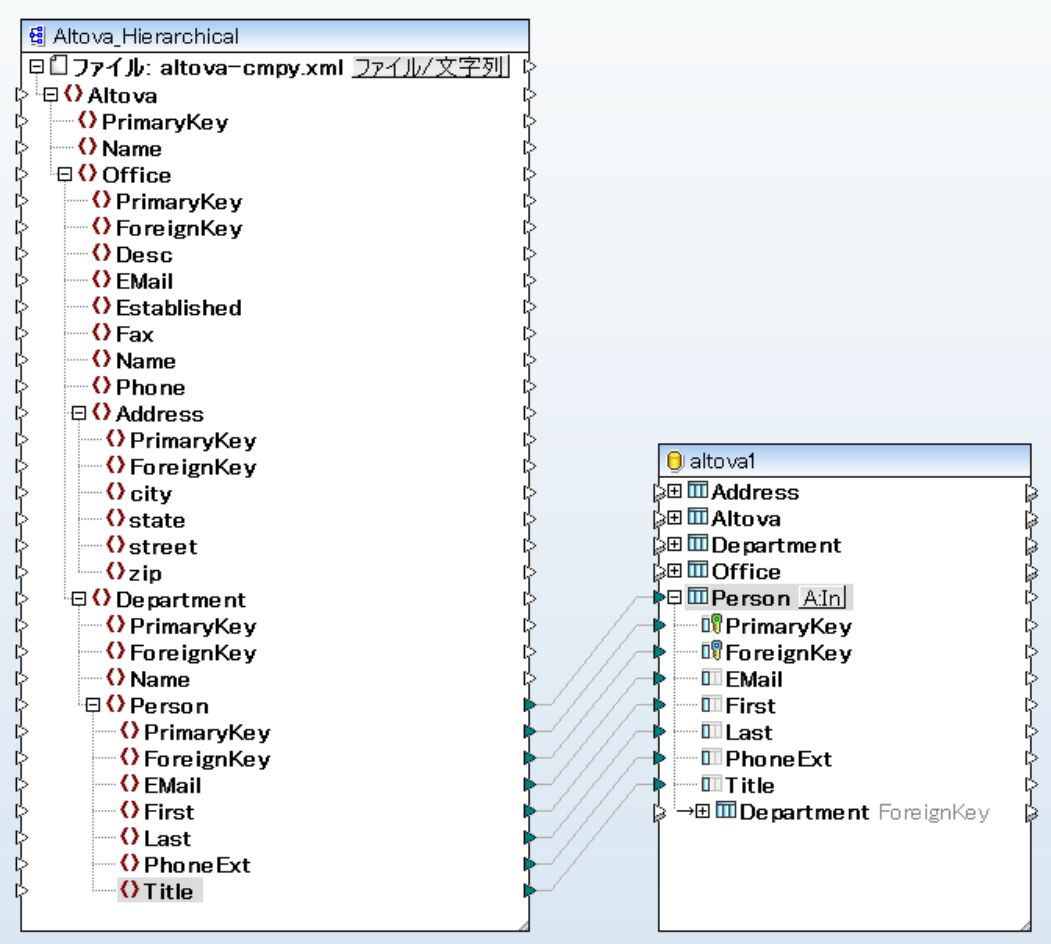
- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml** を参照してください。

ステップ 2：ターゲットデータベースを挿入する

- **「挿入」**メニューから、**「データベース」**をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。 次を参照してください：例："altova.mdb" データベースをマッピングに追加する)。

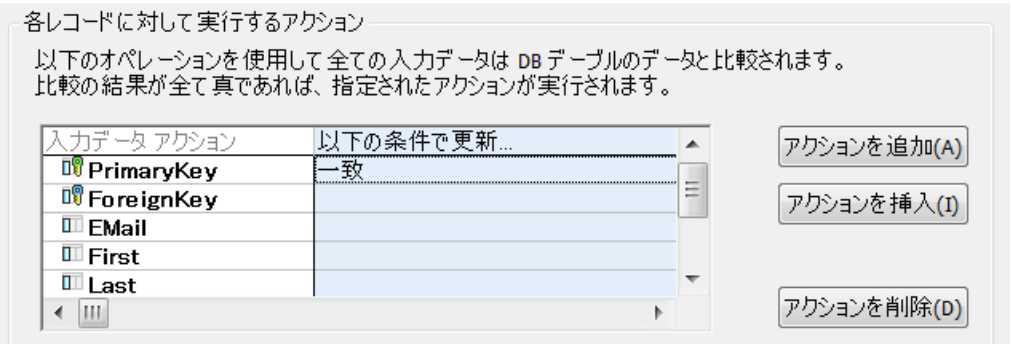
ステップ 3：接続を描く

- 下に表示されるとおりマッピングの接続を描きます。



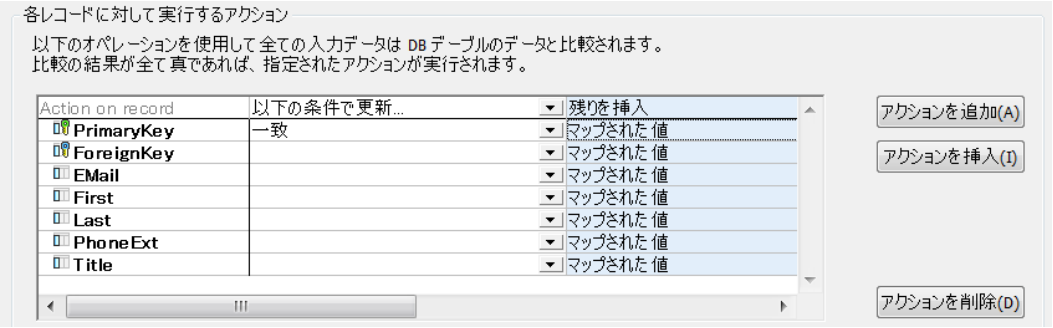
ステップ 4：「以下の条件で更新し、残りを挿入」アクションを構成する

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション：挿入**（ A:In ）ボタンをクリックします。
2. **「レコードに対するアクション」**の横の、**「以下の条件で更新する..」**を選択します。これはデータベーステーブルアクションを条件付きの更新アクションに変更します。つまり、条件が満たされる場合のみ、現在のレコードは更新されます（次のステップを参照してください）。
3. 「PrimaryKey」値の横にあるオプションを、以下に表示されるように「equal」に選択します。つまり、データベースの記録は、「PrimaryKey」値が、マッピングから与えられるの「PrimaryKey」値に等しい場合のみ更新されます。



4. **「アクションの追加」**をクリックします。これは、新規のアクションを既存の**「以下の条件で更新する」**アクションの

右に追加します。 新規のアクションを **残りを挿入** として構成します：



上のイメージでは、データベーステーブルアクションは、マッピングの目的に従い構成されています。つまり、 **以下の条件で更新する**... 条件が満たされる場合のみレコードは更新されます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。

(この例では必要ありませんが)2つ以上のアクションを同じデータベーステーブルに対して定義することもできます。マッピングランタイムでは、アクションは左から右へ実行されます。最後の **挿入** アクションは最後と考えられ、このアクションの後のアクションは無視されます。

ダイアログボックスの **「アクションの追加」** ボタンは、選択されたアクションの **後に** 新規にアクションを追加します。 **「アクションの挿入」** は、新規のアクションを選択されたアクションの **前に** 追加します。既存のアクションを削除するには、アクションをクリックし、を **「アクションの削除」** クリックします。

5. ダイアログボックスを閉じるために **「OK」** をクリックします。マッピングでは、 **アクション:挿入** A:In ボタンが、 **アクション:更新 ;挿入** (A:Up,In) これは、更新と挿入アクションが、このテーブルのために構成されることを示します。

ステップ 5 : マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために **「出力」** タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander',
    [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis',
    [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] = 'Cicada',
    [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
```

データベース更新前のSQL スクリプト 部分的なビュー)

INSERT ステートメントがプレビューのスクリプト内で表示されていないことに気がついているかもしれません。これは、レコードは条件付きで挿入され、INSERT ステートメントは (マッピングが実行される前は未知である) **以下の条件で更新する** アクションの結果 に依存するため、通常の振る舞いです。

メモ: 特定のデータベース型のために、MapForce は、MERGE ステートメントをUPDATE ステートメントの代わりに作成します。詳細に関しては、次を参照してください MERGE ステートメント.

データベースに対してスクリプトを実行するには以下を行います :

- **出力**」メニューから、「**SQL-スクリプトの実行**」をクリックします。

マッピングが実行され、スクリプトがデータベースに適用されました。**出力**」タブ内でINSERT ステートメントが表示されていることにご注意してください。

```
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   The database was connected using the following connection-string:
6   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 row(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander', [PhoneExt]
     = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 row(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis', [PhoneExt] =
     963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 row(s).
17
18  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] = 'Cicada',
    [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
19  -->>> OK. 0 row(s).
20
21  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (30, 1,
    'c.Cicada@microtech.com', 'Camilla ', 'Cicada', 765, 'HR')
22  -->>> OK. 1 row(s).
```

データベース更新後のSQL スクリプト (部分的なビュー)

メモ: MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

MapForce 内のDB クエリタブ内の「Person」テーブルを開くと (次を参照してください:データベースの参照とクエリ)、マッピングの結果は以下の通りです :

- 対応する主キーin XML ファイル内で、対応する主キーを持つすべてのデータベースの記録は更新されました。例は、主キー1、2、3、4、および 5を持つレコードです。
- XML ファイル内に対応するキーをもたない全てのデータベースの記録は、影響を受けず、変更されていません。例は、主キー6、7、8、および 9を持つレコードです。
- 新規のレコードが (データベース内でキーが存在していない箇所である)「Person」テーブルに挿入されました。例は、主キー 30と31を持つレコードです。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	A.Aldrich@microtech.com	Albert	Aldrich	582	Manager
2	2	1	b.bander@microtech.com	Bert	Bander	471	Accounts Receivable
3	3	1	c.clovis@microtech.com	Clive	Clovis	963	Accounting Manager
4	4	2	d.Durnell@microtech.com	Dave	Durnelll	621	Marketing Manager Europe
5	5	2	e.ellas@microtech.com	Eve	Ellas	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	n.newbury@microtech.com	Nira	Newbury	241	Support Engineer
15	15	4	o.origone@microtech.com	Olanda	Origone	345	Support Engineer
16	16	5	s.meier@microtech.com	Paul	Ponetti	114	Office Manager
17	17	5	t.bone@microtech.com	Quang	Qui	331	Accounts Receivable
18	18	6	m.nafta@microtech.com	Rita	Rotter	122	PR & Marketing Manager US
19	19	7	v.bass@microtech.com	Silla	Semua	716	IT Manager
20	20	7	c.franken@microtech.com	Tom	Tinker	147	Support Engineer
21	21	7	m.redgreen@microtech.com	Uri	Ugoner	152	Support Engineer
22	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースを更新後の「Person」テーブル

7.2.4.5 MERGE ステートメント

データベーステーブルを更新し、データを挿入する特定のマッピングに関しては、MapForce は、データベースに対して実行されるMERGE ステートメントをマッピングランタイムも生成する可能性があります。(次も参照してください: [以下の条件で更新し、残りを挿入」アクション])。実行エンジンがMapForce である必要は必ずしもありません。次を参照してください: [データベースを変更するマッピングの実行]。

MERGE ステートメントは、次のデータベース型によりサポートされています:

- SQL Server 2008 および以降
- Oracle
- DB2
- Firebird

INSERT とUPDATE ステートメントを1つにまとめるため、MERGE ステートメントは、データベースサーバーの呼び出しの回数を減らします。また、MERGE ステートメントの場合、整合性は、データベースによりチェックされます。データベース型を検出すると、MapForce は、自動的にMERGE ステートメントを作成します。MapForce によるMERGE ステートメントの生成を手動で影響することはできません。

マッピングがランタイムにデータベースMERGE ステートメントを実行するかを確認します (NSERT とUPDATE ステートメントの組み合わせの適用に対して):

1. **「以下の条件で更新する..」**と **「残りを挿入」**アクションを使用するマッピングを作成します。例えば、次を参照

してください：[以下の条件で更新し、残りを挿入」アクション](#)。
2. マッピングをプレビューするには、 **「出力」** タブをクリックします。

MERGE がデータベース型によりサポートされている場合、生成されたSQL スクリプトは、MERGE ステートメントを含みます。例：

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=VIETSQL14\VIETSQL14;Persist Security Info=True;Provider=SQLOLEDB.1
6   */
7
8   SET QUOTED_IDENTIFIER ON
9
10  SET ANSI_NULLS OFF
11
12  MERGE INTO [Production].[Location] AS T USING ( VALUES ( NULL, (CAST('1' AS nvarchar(50))) ) ) AS S ( [LocationID],
    [Name] ) ON ( (S.[LocationID] = T.[LocationID]) ) WHEN MATCHED THEN UPDATE SET [Name] = S.[Name] WHEN NOT
    MATCHED THEN INSERT ( [Name] ) VALUES ( S.[Name] );
13
```

データベース型がMERGE をサポートしない場合、生成されたSQL スクリプトは、UPDATE ステートメントのみを含みます。 **「以下の条件で更新する..」** 条件が満たされている場合のみ、実行されるため、(マッピングの実行前には未知である)INSERT ステートメントは、プレビューでは表示されません。

制限：

- ステートメントに関しては、「バルク転送」オプション (次を参照してください：[バルク転送 (MapForce Server)](#)) は、ODBC とJDBC データベース接続のためにのみによりサポートされています。

7.2.4.6 親テーブルを更新する際の子テーブルのためのオプション

マッピングは、「親」テーブルであるテーブルを更新する際、 (つまり、テーブルに対して外部キー リレーションシップ持っている場合)、ソースデータとターゲットテーブル内で、どのように依存するレコードが扱われるかを構成することができます。例えば、**altova.mdb** データベース内の「Department」テーブルを更新するとします。すべての個人 は、外部キーを使用して、department に、リンクされています。「Person」テーブルに対しても、挿入、更新、または削除などの アクションを取る必要があるかもしれません。これを行うことは、データベースの整合性を保持し、マッピングエラーを回避する助けとなります。

このトピックは、親 「Department」テーブルを更新した際に、 「Person」テーブルに対して使用することのできるオプションについて説明しています。以下のサンプルファイルが使用されています：

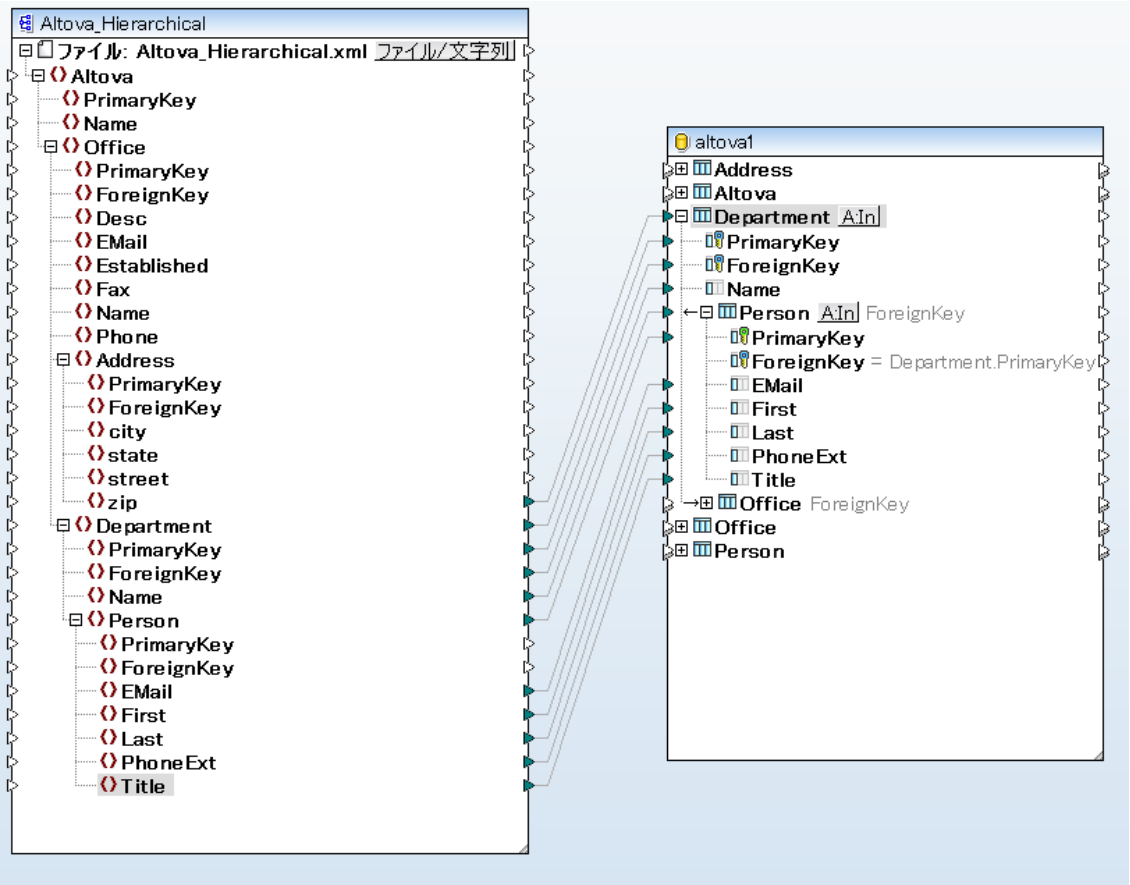
- **altova-cmpy-extra.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** ？ 更新されるターゲットデータベースです。

全てのファイルは **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください[データベースを変更するマッピングの実行](#)。

最初に、ソースXML スキーマとインスタンスとターゲットデータベースをマッピングに追加します (次を参照：[例："altova.mdb" データベースをマッピングに追加する](#))。 [以下の条件で更新し、残りを挿入」アクション](#)と同じステップを踏

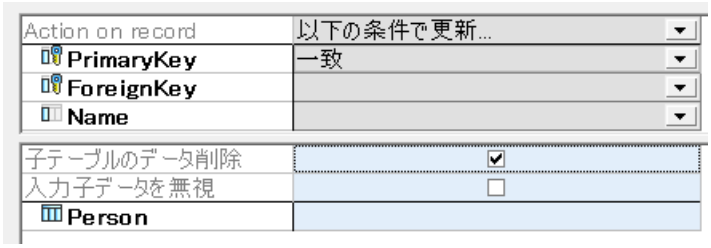
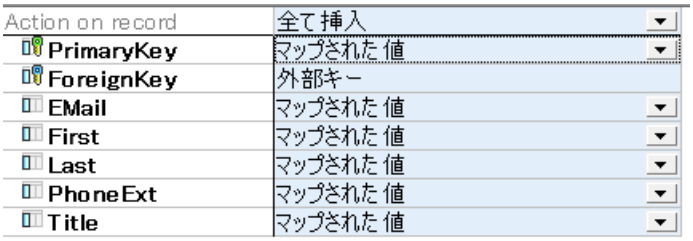
んでください。次に、以下に表示されるようにマッピングの接続を描きます：



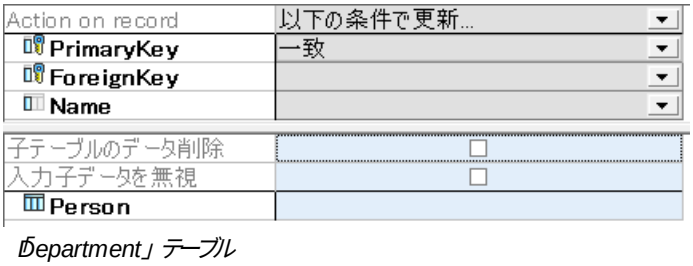
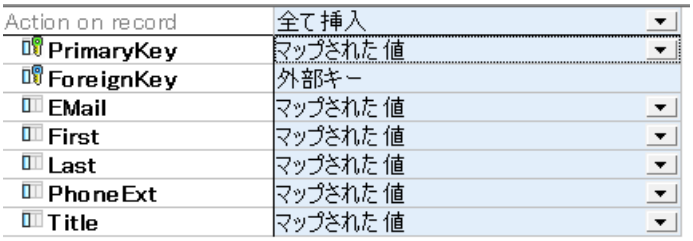
上記のように、マッピングは、ターゲットデータベース内の「Department」テーブルを更新します。「Department」テーブルは、「ルート」テーブルとして選択されています。ルートテーブルに関する情報と、その重要性に関する詳細は、次を参照してください：データベースリレーションシップの処理。子「Person」テーブルに対して取られるアクションは、このトピックの主題です。

次のテーブルは、多種の構成オプションと対応するマッピングの結果を表しています。これらのオプションは、親「Department」テーブルと子「Person」テーブルのデータベーステーブルアクションダイアログボックスから選択することができます。

構成 A

設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の`PrimaryKey` に対応する箇所で Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存の Department レコードを更新しません。 • データベース内で既存ではないすべての Person レコードを、入力 XML インスタンスから挿入します。 • **『以下の条件で更新する..』**条件を満たす Department レコードの子データ（Person レコード）を削除します。

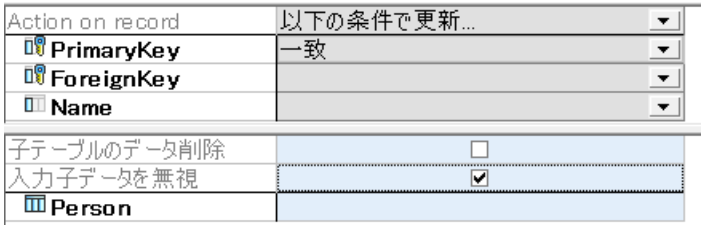
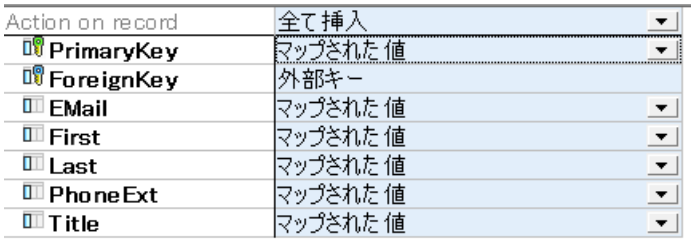
構成 B

設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	SQL 実行エラーと共にマッピングは失敗します。理由は、マッピングが、新規の Person レコードを既存の Person レコードと同じ主キーを使用して挿入しようと試みたからです。入力 XML からのレコードを、データベース内に既存のレコードに加えて追加する場合は、次を参照してください。

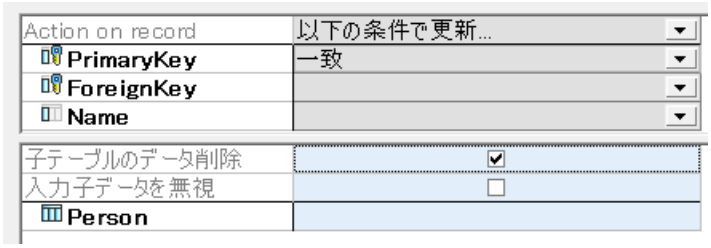
構成 C

設定	マッピングの結果
 「Department」テーブル  「Person」テーブル	• ソースXML 内のPrimaryKey がデータベーステーブル内のPrimaryKey に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない (このようなキーはソースに存在しません) 既存のDepartment レコードを更新しません。 • 新規のPerson レコード (生成された主キーと共に) は、既存のものに加え、Person テーブルに挿入されます。

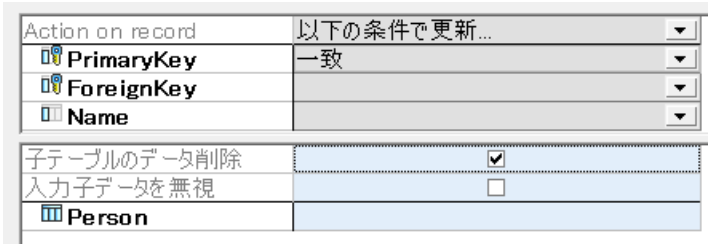
構成 D

設定	マッピングの結果
 「Department」テーブル  「Person」テーブル	• ソースXML 内のPrimaryKey がデータベーステーブル内のPrimaryKey に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない (このようなキーはソースに存在しません) 既存のDepartment レコードを更新しません。 • オプション**入力子データを無視する**が親 Department テーブルのために有効化されているため、Person テーブルにレコードは挿入されません。

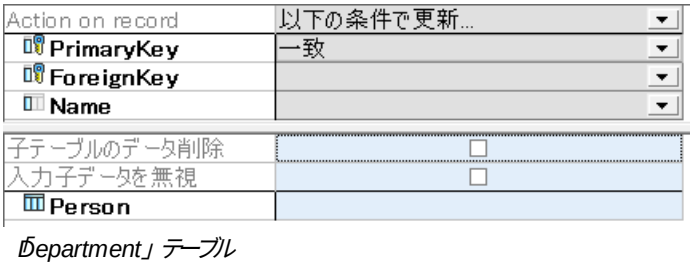
構成 E

設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の`PrimaryKey` に対応する箇所で Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存の Department レコードを更新しません。 • ソースXML 内に対応する`PrimaryKey` をもつ Department にリンクされているレコードを全て削除します。理由は、親 Department テーブルのために **「子テーブル内のデータの削除」**オプションが有効化されているからです。 • **「以下の条件で更新する..」**条件を満たさなかった部署にリンクされているPerson レコードは、データベースに残されます。 • Person テーブル内のレコードは更新されていません。

構成 F

設定	マッピングの結果
 「Department」テーブル  「Person」テーブル	• ソースXML 内のPrimaryKey がデータベーステーブル内のPrimaryKey に対応する箇所でDepartment レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない (このようなキーはソースに存在しません) 既存のDepartment レコードを更新しません。 • ソースXML 内に対応するPrimaryKey をもつDepartment にリンクされているレコードを全て削除します。この理由は、**『子テーブル内のデータの削除』**オプションは、親 Department テーブルのために有効化されているからです。 • **『以下の条件で更新する..』**条件を満たさなかったDepartment にリンクされているPerson レコードは、データベースに残されます。

構成 G

設定	マッピングの結果
 「Department」テーブル  「Person」テーブル	• ソースXML 内のPrimaryKey がデータベーステーブル内のPrimaryKey に対応する箇所でDepartment レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない (このようなキーはソースに存在しません) 既存のDepartment レコードを更新しません。 • 次の2つの条件を満たす、すべてのPerson レコードを削除します: a. 対応するソースXML 内に対応するPrimaryKey が存在するDepartment にリンクされているPerson レコード。 b. 対応するソースXML 内に対応するPrimaryKey が存在するPerson レコード

7.2.4.7 「以下の条件で削除 ...」アクション

テーブルアクション**Delete if...** は、データベーステーブルからデータを条件付きで削除する場合に使用されます。データベーステーブル アクション ダイアログボックスからこのアクションを定義することができます。 (次を参照してください:データベーステーブルアクション設定) 例えば、ソースからのデータXML をターゲットデータベースにマッピングする際、ソースXML 内の特定の条件が、ターゲットデータベースの記録 (通常、主キーの値) 内のフィールドに対して同じかをチェックするために、**「以下の条件で削除 ..」**条件を構成することができます。 **「以下の条件で削除 ..」**条件がtrue の場合、(つまり、2つのフィールドが等しい場合) データベースの記録は、マッピングが実行されると削除されます。

メモ: **「以下の条件で削除 ..」**テーブルアクションをデータベース テーブル アクション ダイアログボックス内で使用することのできる **「子テーブル内のデータの削除」**オプションと混合しないようご注意してください。 **「以下の条件で削除 ...」**テーブルアクションは、アクションが定義されているテーブルのみに影響を与え、他のテーブルには影響はありません。

この例は、データベース テーブルからデータを条件付きで削除する方法、また削除条件の結果をベースにして同じデータベーステーブルにレコードを挿入する方法を説明します。

この例は以下のファイルを使用します :

- **altova-cmpy-extra.xml** ? データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ? 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** ? 更新されるターゲットデータベースです。

全てのファイルは **<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください データベースを変更するマッピングの実行。

このマッピングの目的は以下の通りです :

- 同じ`PrimaryKey` を持つ、person レコードが、ソースXML とターゲットPerson テーブル内両方に存在する場合、これらはPerson テーブルから削除されなければなりません。
- ソースXML からの全てのレコードは、Person テーブルに挿入される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1 : ソース XML コンポーネントを挿入する

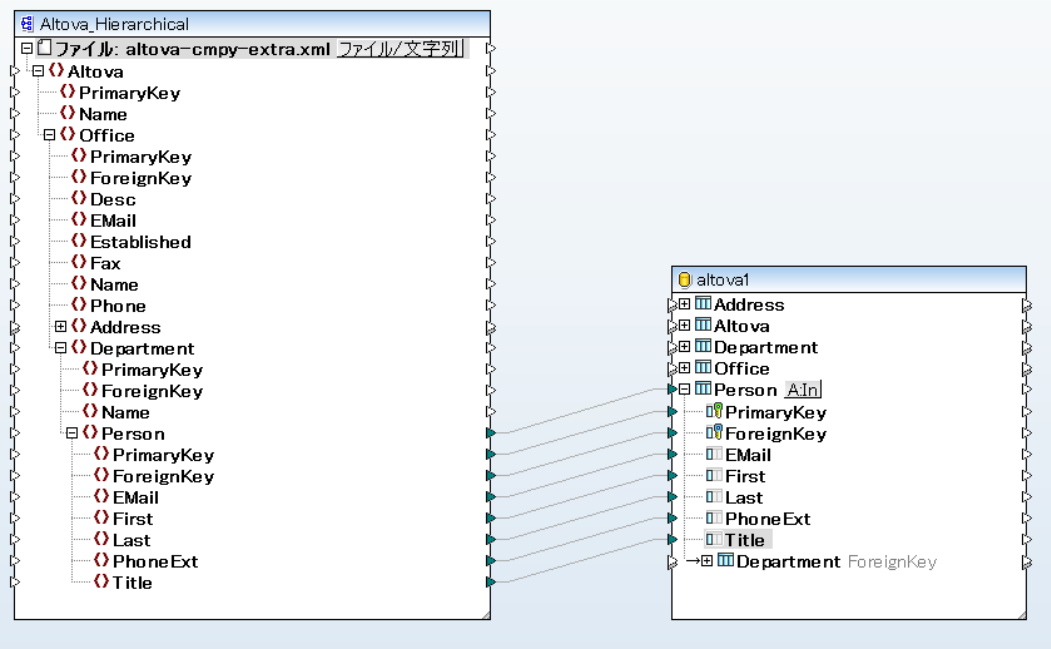
- **「挿入」**メニューから、**「XML スキーマ/ファイル」**をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml** を参照してください。

ステップ 2 : ターゲットデータベースを挿入する

- **「挿入」**メニューから、**「データベース」**をクリックして、 **altova.mdb** に接続するためにウィザードのステップを行ってください。 (次を参照してください:例 : "altova.mdb" データベースをマッピングに追加する)

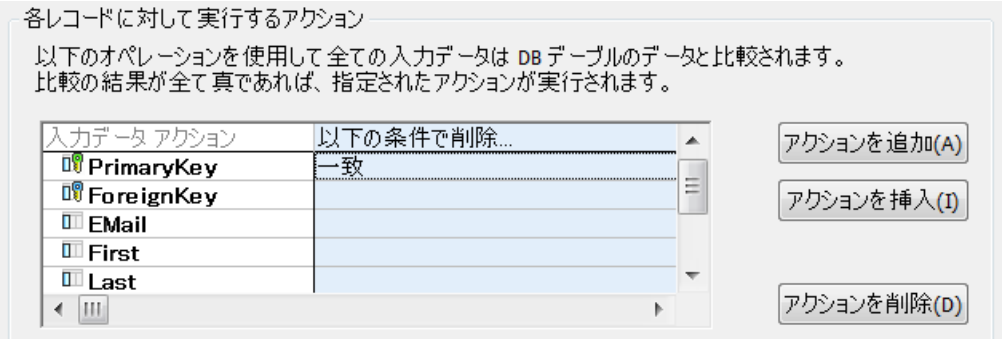
ステップ 3：接続を描く

- 下に表示されるとおりマッピングの接続を描きます。

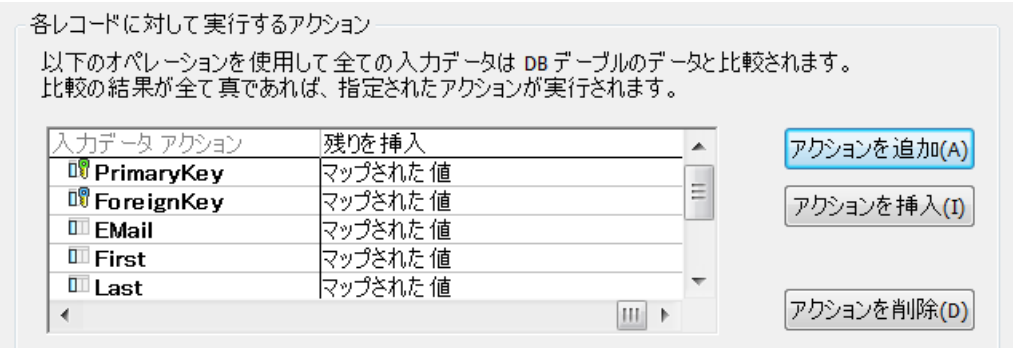


ステップ 4： 以下の条件で削除 ..」 残りを挿入」" アクションを構成する

1. ターゲットコンポーネントで、 「Person」テーブルの横の **アクション：挿入** (A:In) ボタンをクリックします。
2. **「レコードに対するアクション」**の横の、 **以下の条件で削除 ..」**を選択します。これは、データベーステーブルアクションを条件月の 「アクションの削除」に変更します。つまり、条件が満たされる場合のみ、現在のレコードは削除されます（次のステップを参照してください）。
3. **「PrimaryKey」**の横の、値 **「一致」**を選択します。これにより、更新条件を定義します。つまり、下に表示されるとおり、データベースの記録は、 **「PrimaryKey」**値が、マッピングより与えられる **「PrimaryKey」**値に等しい場合、削除されます。



4. **「アクションの追加」**をクリックします。これは、新規のアクションを既存の **「以下の条件で削除 ..」**アクションの右に追加します。新規のアクションを **「残りを挿入」**として構成します ：



上のイメージでは、データベーステーブル アクションは、マッピングの目的に従い構成されています。つまり、**Delete If**... 条件が満たされる場合のみ、レコードは削除されます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。

5. ダイアログボックスを閉じるために **「OK」**をクリックします。マッピングの背景で、**アクション:挿入** A:In ボタンが、**アクション:削除 ;挿入** (A:De,In) ボタンに変更されていることにご注意してください。これは、削除と挿入アクションが、このテーブルのために構成されることを示します。

ステップ 5 : マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために **「出力」**タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。このスクリプトはプレビューのためであり、データベースをまだ変更していません。データベースに対してスクリプトを実行するには以下を行います :

- **「出力」**メニューから、**「SQL-スクリプトの実行」**.をクリックします。

メモ: MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください :データベースを変更するマッピングの実行。

MapForce 内の DB クエリダイアログ内の「Person」テーブルを開くと (次を参照してください :データベースの参照とクエリ)、マッピングの結果は以下の通りです :

- XML ファイル内に対応する主キーを持つすべてのデータベースの記録は、削除されました。例は、主キー 1、2、3、4、および 5 を持つレコードです。
- XML ファイル内に対応するキーをもたない全てのデータベースの記録は、影響を受けず、変更されていません。例は、主キー 6、7、8、9、10、11、12、および 13 を持つレコードです。
- 新規のレコードが (データベース内でキーが存在していない箇所である)「Person」テーブルに挿入されました。例は、主キー 30 と 31 を持つレコードです。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
2	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
3	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
4	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
5	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
6	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
7	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
8	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
9	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
10	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースを更新後の「Person」テーブル

7.2.4.8 *以下の条件で無視 ..」アクション*

テーブルアクション **以下の条件で無視 ..」**は、定義された条件に従い、データベース内の特定のレコードが更新されることを回避するために使用されます。 **以下の条件で無視 ..」**アクションは、（**残りを挿入」**アクションなどの）他のデータベーステーブルアクションと組み合わせて使用される場合のみ、意味があります。例えば、ソースXML からのデータをターゲットデータベースにマッピングする場合、 **以下の条件で無視 ..」**条件を構成し、ソースXML 内の特定のフィールドがターゲットデータベースの記録内の（通常、主キーの値である）フィールドと等価であるかをチェックするように構成することができます。 マッピングが実行される際、 **以下の条件で無視 ..」**条件がtrue の場合、（つまり、2つのフィールドが等しい場合）、データベースの記録は無視され、次に定義されたアクション（例えば、 **残りを挿入」**）は、実行されます。

この例は、 **以下の条件で無視 ..」**条件の結果に基づいたデータベーステーブルにレコードを挿入する方法について説明します。この例は、次のファイルを使用します：

- **altova-cmpy-extra.xml** ？ データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** ？ 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** ？ 更新されるターゲットデータベースです。

全てのファイルは**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください[データベースを変更するマッピングの実行](#)。

このマッピングの目的は以下の通りです：

- ソースXML とターゲットPerson テーブル内の両方で、同じ`PrimaryKey` を持つ個人のレコードが存在する場合、 アクションは取られません。（つまり、無視されます）。
- 個人のレコードがPerson テーブル内に存在する上記の条件を満たさない場合、アクションは取られません。
- Person テーブル内に対応する（主キーが無い）ソースXML からのレコードは、新規として扱われ、新規の主キーと共にPerson テーブルに挿入されます。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1：ソース XML コンポーネントを挿入する

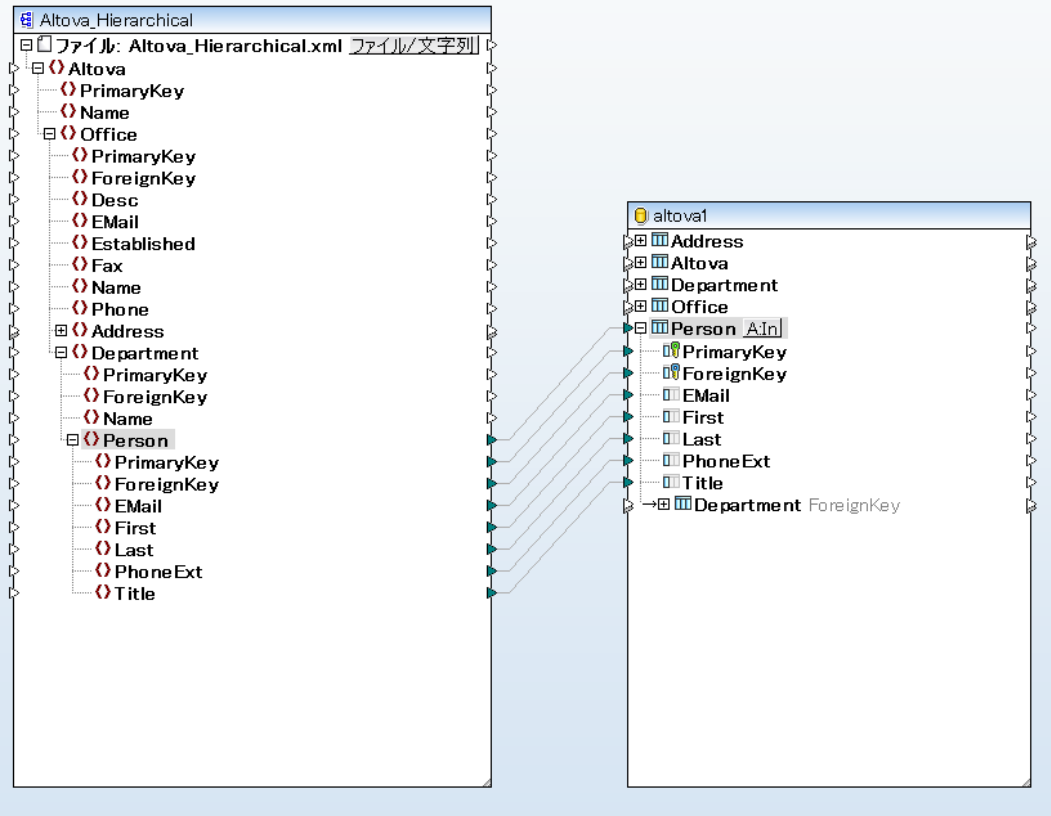
- **挿入**」メニューから、**「XML スキーマ/ファイル」**をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml** を参照してください。

ステップ 2：ターゲットデータベースを挿入する

- **挿入**」メニューから、**「データベース」**をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。（次を参照してください：例："altova.mdb" データベースをマッピングに追加する）。

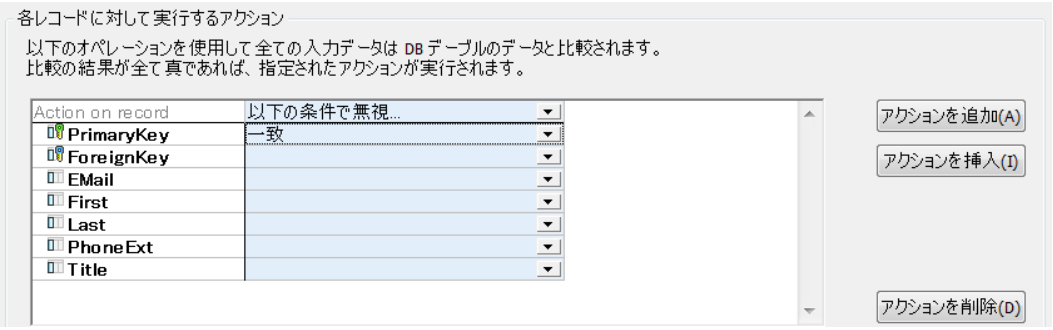
ステップ 3：接続を描く

- 下に表示されるとおりマッピングの接続を描きます。

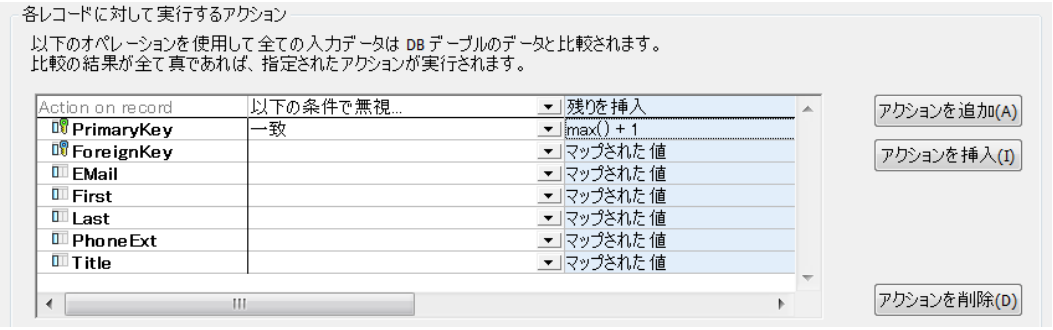


ステップ 4：" 「以下の条件で無視 ..残りを挿入」" アクションを構成する

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション：挿入**（A:In）ボタンをクリックします。
2. **「レコードに対するアクション」**の横の、**「以下の条件で無視 ..」**を選択します。これは、条件付きの無視アクションへのデータベーステーブルアクションを変更します。つまり、条件が満たされる場合のみ、現在のレコードは無視されます（次のステップを参照してください）。
3. **「PrimaryKey」**値の横から下に表示されるとおり、**equal** を選択します。これは無視の条件を定義します。つまり、データベースの記録は、`PrimaryKey` 値がマッピングから与えられる`PrimaryKey` の値と同じ場合、無視されます。



4. **「アクションの追加」**をクリックします。これは、新規のアクションを既存の **「以下の条件で無視 ..」**アクションの右に追加します。主キーが**max() + 1** に設定され、下に表示されるとおり、新規のアクションが **「残りを挿入」**として構成されます：



上のイメージでは、データベーステーブルアクションは、マッピングの目的に従い構成されています。つまり、**「以下の条件で無視 ..」**条件が満たされると、レコードはスキップされます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。オプション**max() + 1** は、レコードのために一意の、新規の主キーの値を生成します。

5. ダイアログボックスを閉じるために **「OK」**をクリックします。マッピングでは、**アクション:挿入** A:In ボタンは、**アクション:無視 ;挿入** ( ) ボタンに変更されます。これは、アクションの無視と挿入がこのテーブルのために構成されていることを示しています。

ステップ 5：マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために **「出力」**タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。このスクリプトはプレビューのためであり、データベースをまだ変更していません。データベースに対してスクリプトを実行するには以下を行います：

- **「出力」**メニューから、**「SQL-スクリプトの実行」**をクリックします。

メモ： MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください：データベースを変更するマッピングの実行。

MapForce 内の DB クエリダイアログ内の「Person」テーブルを開くと（次を参照してください：データベースの参照とクエリ）、マッピングの結果は以下の通りです：

- XML ファイル内に対応する主キーを持つ全てのデータベースの記録は、**「以下の条件で無視 ..」**条件を満たし、影響を受けず、変更されていません。例は、主キー 1、2、3、4、および 5 を持つレコードです。
- XML ファイル内で対応するキーを持たない全てのデータベースの記録は、**「以下の条件で無視 ..」**条件を満たしませんでしたが、影響を受けず変更されていません。例は、主キー 6、7、8、9、10、11、12、および 13 を持

つレコードです。

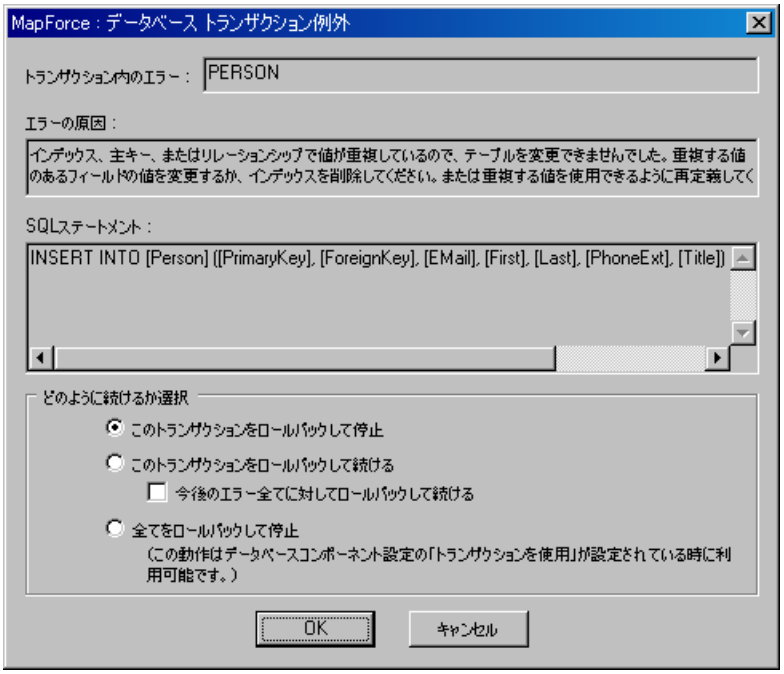
- 新規のレコードが（データベース内でキーが存在していない箇所である）「Person」テーブルに挿入されました。例は、主キー 30 と 31 をソース XML ファイル内に持つレコードです。これらは、それぞれ新規の主キー 22 と 23 と共にデータベースに挿入されます。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	v.callaby@nanonull.com	Vernon	Callaby	582	Office Manager
2	2	1	f.further@nanonull.com	Frank	Further	471	Accounts Receivable
3	3	1	l.matise@nanonull.com	Loby	Matise	963	Accounting Manager
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread	621	Marketing Manager Europe
5	5	2	s.sanna@nanonull.com	Susi	Sanna	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	j.band@nanonull.com	Jessica	Bander	241	Support Engineer
15	15	4	l.king@nanonull.com	Lui	King	345	Support Engineer
16	16	5	s.meier@nanonull.com	Steve	Meier	114	Office Manager
17	17	5	t.bone@nanonull.com	Theo	Bone	331	Accounts Receivable
18	18	6	m.nafta@nanonull.com	Max	Nafta	122	PR & Marketing Manager US
19	19	7	v.bass@nanonull.com	Valentin	Bass	716	IT Manager
20	20	7	c.franken@nanonull.com	Carl	Franken	147	Support Engineer
21	21	7	m.redgreen@nanonull.com	Mark	Redgreen	152	Support Engineer
22	22	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	23	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースの更新後の「Person」テーブル

7.2.4.9 トランザクションロールバックの使用

トランザクションロールバックにより、データベースの例外が発生すると、取られるアクションを決定することができます。具体的には、MapForce から SQL スクリプトを実行しようと試み、エラーが発生すると、下に表示されるダイアログボックスが開かれ、継続方法について問われます。例外が発生する前に、既に実行され成功している SQL ステートメント（トランザクション）が存在可能性もありますので、以前実行されたトランザクションすべてをロールバックするか、または、現在のトランザクションのみをロールバックすることが選択できます。



データベース トランザクション例外 ダイアログボックス

失敗したデータベースアクションをロールバックするには、トランザクションを最初に有効化する必要があります。データベース トランザクションは以下のように有効化することができます :

- データベース レベルで。データベース レベルでトランザクションを有効化するには、データベース コンポーネント設定ダイアログボックスから **「トランザクションの使用」** チェックボックスを選択します (次も参照してください : データベース コンポーネント設定)。
- 個別のテーブルのために。個別のテーブルのためにトランザクションを有効化するには、データベース テーブル アクション ダイアログボックスから **「トランザクションの使用」** チェックボックスを選択します。 (次も参照してください : データベース テーブル アクション設定)。

データベース レベルのトランザクションを、テーブル レベルで有効化することなく、有効化することができます。また、その逆も可能です。データベースの例外が発生すると、使用することのできるロールバック オプションは、どのようにトランザクションが有効化されているかにより異なることにご注意してください :

- トランザクションが有効化されておらず、エラーが発生すると、実行はエラーが発生する箇所で中止されます。以前に成功したすべての SQL ステートメントが実行され、結果はデータベース内に保管されます。トランザクションをロールバックすることは不可能です。
- トランザクションが、データベース レベルではなく、テーブル レベルのみで有効化されている場合、実行はエラーが発生する箇所で中止されます。この場合、オプション **「ロールバックして停止する」** が無効化されています。子の特定のテーブルのための現在のトランザクションのみをロールバックし、マッピングの実行を続行するか、マッピングを完全に停止することができます。
- トランザクションがデータベース レベルのみで有効化されている場合、実行はエラーが発生する箇所で中止されます。以前に成功したすべての SQL ステートメント ロールバックされます。データベースに変更は付け加えられません。
- トランザクションが、データベース レベルとテーブル レベルの両方で有効化されている場合、実行はエラーが発生する箇所で中止されます。この場合、 **「ロールバックして停止する」** を選択し、データベース全体のために以前成功した SQL ステートメントをロールバックします。最後のトランザクションのみをロールバックし、マッピングの実行を継続するには、 **「このトランザクションをロールバックして継続する」** を選択してください。

データベース トランザクション例外 ダイアログボックスは、 **「SQL | SQL-Script を実行」** メニュー コマンドを使用して、MapForce から直接マッピングを実行された場合のみ表示されることにご注意してください。マッピングが生成されたコード、

または、MapForce Server など)他の環境で実行された場合、、そしてエラーが発生した場合、自動的にエラーのあったトランザクションのロールバックが発生します。この場合、以前に実行され、成功したSQL ステートメントは、上記の理由で、データベースレベルのトランザクションが有効化されている場合、ロールバックされます。

「キャンセル」ボタンを押下することで、SQL ステートメントの実行がロールバックされが停止します。

7.2.4.10 バルク挿入 (MapForce Server)

「バルク転送」コマンドによりMapForce コンポーネント (TXT、CSV、DAT など)からデータベーステーブルに高速にデータを挿入することができます。このオプションを使用することにより、1つのステートメントを実行することにより、挿入プロセスを劇的にスピードアップすることができます。

バルク転送は以下の条件が満たされた場合にサポートされます:

- マッピング変換言語がBUILT-IN に設定されている場合。詳細に関しては、次を参照してください:変換言語の選択。
- マッピングが (スタンドアロン、または、FlowForce Server の管理下で)MapForce Server により作動している場合。これは、マッピングが .mfx フォーマットにコンパイルされるか、または、FlowForce Server にデプロイされる必要があることを意味します。詳細に関しては、次を参照してください:MapForce Server 実行可能ファイルにマッピングをコンパイルするとFlowForce Server にマッピングをデプロイする。
- MapForce Server ライセンスは、マルチコアマシン上の "単一スレッドの実行 "だけに制限されません。つまり、チェックボックスAltova LicenseServer の "サーバー管理 "タブ内の **「単一スレッドの実行に制限する」**が無効化されている必要があります。
- データベースアクションは 全て挿入する」です。
- データがバルクロードされるテーブルは、 "リーフテーブル "である必要があります。すなわち、データベースの一番下の階層である必要があります。関連したテーブル、ビュー、またはマッピング内のテーブルを参照するストアドプロシージャは存在してはなりません。
- データベースドライバーはWHERE 条件でのバルク挿入をサポートします。

メモ: ODBC によりMicrosoft Access とMySQL に接続している場合、バルク転送はODBC ドライバーによりサポートされていません。

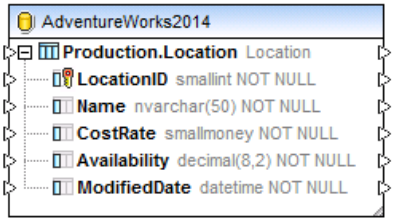
この例は、サンプル**source.txt** ファイルからデータをターゲットデータベースにバルクロードするマッピングを作成する方法について説明しています。例は、SQL Server 2014 とAdventureWorks 2014 データベースを使用しています。後者は、CodePlex Web サイトからダウンロードすることができます (https://msftdbprodsamples.codeplex.com/)。

```
Location A,15.3,39
Location B,46,34
Location C,56.33,0
Location D,0,399
Location E,0,97.43
```

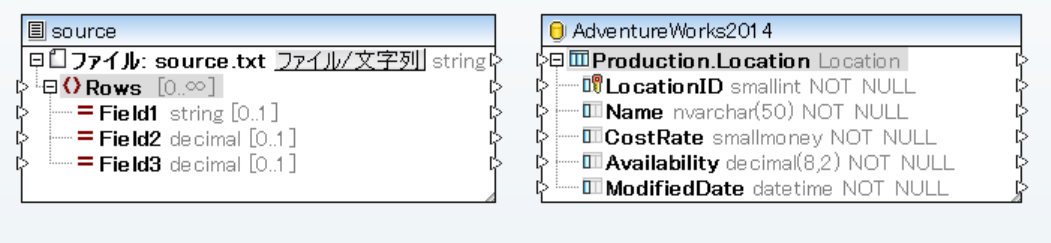
source.txt

バルク転送の定義:

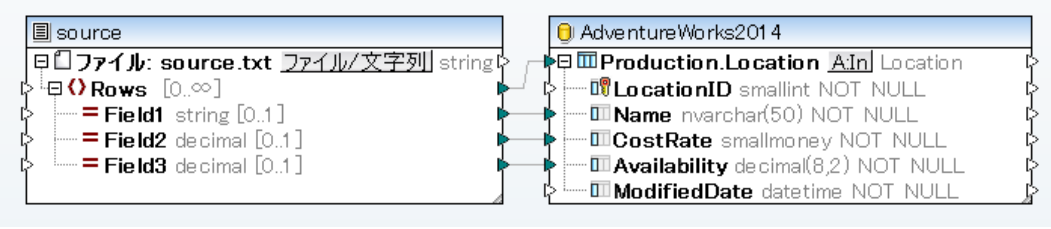
1. 内蔵の ターゲット出力アイコン BUILT IN が有効化されていることを確認してください。
2. AdventureWorks 2014 データベースに接続し、 "Production.Location" テーブルをマッピングに追加します。詳細に関しては、次を参照してください:データベースをマッピングに追加する。



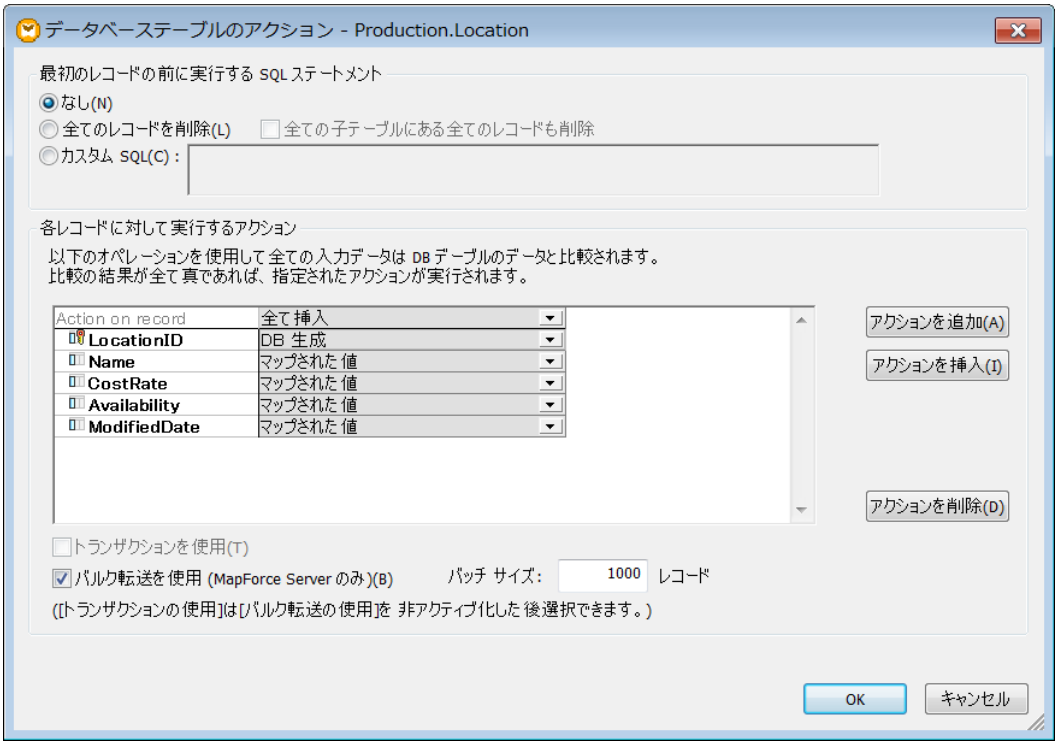
3. **挿入**」メニューから、**「テキスト」**をクリックして、(上に表示されている**source.txt** サンプル等の)ソーステキストファイル をマッピングに追加します。詳細に関しては、次を参照してください[CSV とテキストファイル](#)。ソースとターゲット コンポーネントのデータ型が同じであることを確認してください。 **「データ型の表示」**( ) ツールボタンが有効化されている場合、データ型はコンポーネント上で表示されます。



4. 下に表示されるとおりマッピングの接続を描きます。**データベースアクション**ボタン A:In は、テーブル名の右に表示されていることに注意してください。



5. **データベースアクション**ボタン (A:In)をクリックして、"バルク転送の使用 "チェックボックスを有効化します。**「OK」**をクリックして確認します。 **「バルク転送の使用」**チェックボックスを選択して、 **「OK」**をクリックして確認します。



上のダイアログボックス内で、「レコードに対するアクション」が「全て挿入」に設定されていることを確認してください。**「バッチサイズ」**フィールドは、各アクションに挿入されるレコードの数量を定義します。

メモ: **「バルク転送の使用」**オプションが有効化されていると、**「トランザクションの使用」**オプションは無効化されます。また、**「トランザクションの使用」**オプションが有効化されていると、**「バルク転送の使用」**オプションは無効化されます。トランザクション処理を有効化するには、**「バルク転送の使用」**チェックボックスのチェックを解除します。

バルク挿入が有効化された次のステップは、MapForce Server 内でのマッピングの実行です (スタンドアロン、または、FlowForce Server の管理下で)。詳細に関しては、次を参照してください MapForce Server 実行可能ファイルにマッピングをコンパイルする と FlowForce Server にマッピングをデプロイする。

7.2.4.11 データベーステーブルアクション内の Null 値の処理

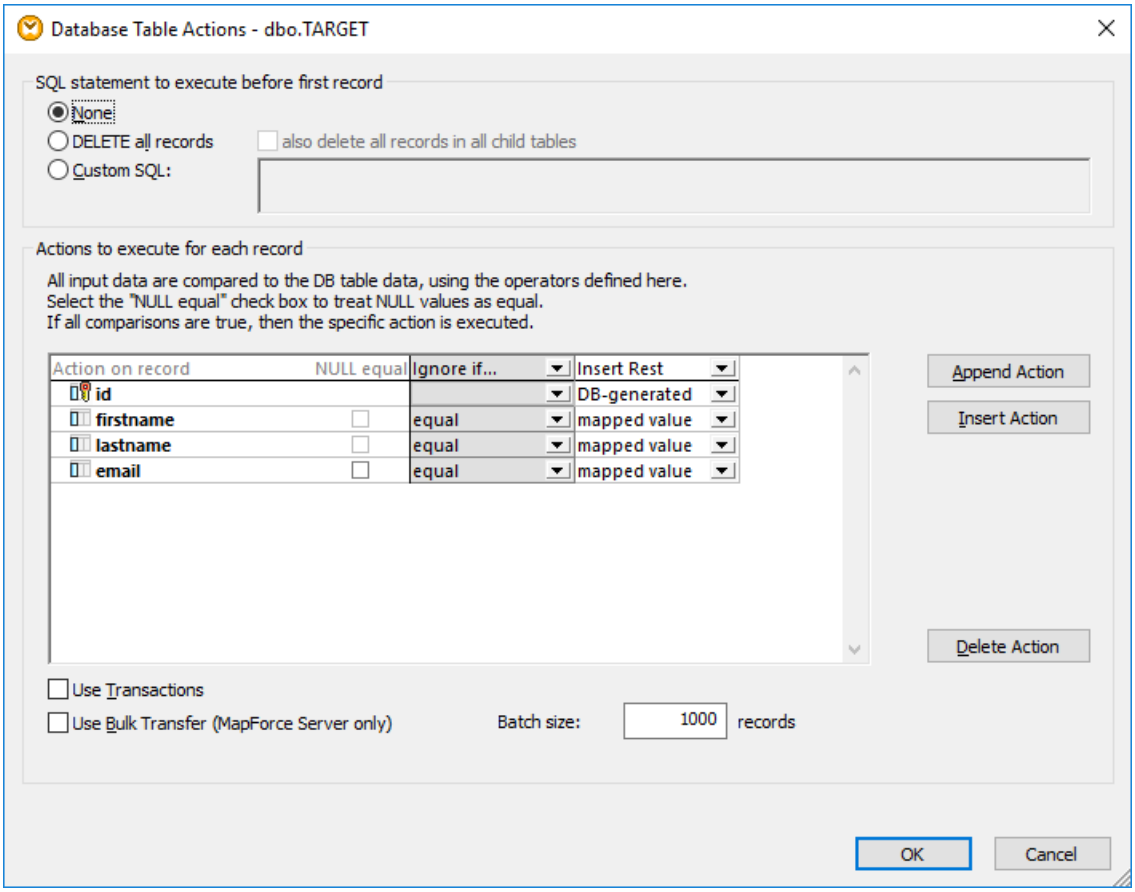
「Ignore If」、「Update If」、「Delete If」などのテーブルアクションを使用してマッピングがターゲットデータベースを更新する場合、MapForce はソースデータをターゲットデータに対して比較し、結果として内部データベース更新クエリを生成します。(これらの内部クエリは、MapForce の**出力**ペイン内のプレビューのために使用することができます次を参照してください:データベースを変更するマッピングの実行)。生成されたクエリは、「データベーステーブルアクション」ダイアログボックスから定義された比較条件を反映しています (次を参照:データベーステーブルアクションの設定)。

Null の比較は、SQL とデータベースのコンテキスト内の複雑な課題です。異なるデータベースの型全体で Null の値を比較するための共通の方法は存在しません。マッピングに含まれるデータベースの型が受け入れられるルールに基づき MapForce パースペクティブから、NULL に対応した方法でデータを比較するようにデータベースマッピングを構成することは可能です。「NULL-対応」とは、データの比較のスコープのために NULL 値が扱われることを意味します (それ以外の場合、マッピングが期待しない結果が生成される可能性があります)。NULL-対応 は以下の条件が満たされると有効化することができます:

1. 「データベーステーブルアクション」ダイアログボックスは、「Ignore if」、「Update if」、「Delete if」アクションを含んでいます。

2. これらのアクションは、NULL 値を含んでいる可能性のあるレコードに対して行われます。
3. ソーステーブル内のNULL 値は、ターゲットテーブル内のNULL 値の等価として扱われる必要があります。

デフォルトでは、NULL-対応は、無効化されています。条件が満たされており、NULL-対応が無効化されている場合、ターゲットデータベーステーブルが期待通り更新されないインスタンスが存在する場合があります（例えば、必要以上の行が挿入、または、更新されます）。NULL 値 がデータの比較を影響し、希望しない結果を生成するためにこのような状況が発生します。このような状況を回避するために、「データベーステーブルアクション」ダイアログボックスから各 Null 許容 フィールド (下のイメージ内 email) の横のチェックボックスを選択します。Null 許容であるフィールドのためにのみ、また、少なくとも1つのテーブルアクションが「等価」または「等価（大文字と小文字を区別しない）」場合のみ、チェックボックスを選択することができます。



データベーステーブルアクション ダイアログボックス

サンプル

マッピング内のNULL 対応を良く理解するために、NULL データの比較が起こるサンプルを分析してみましょう。このサンプルは、Microsoft SQL Server データベースを使用します。しかしながら、サポートされる他のデータベースの型にも適用することができます。任意で、Microsoft SQL Server を使用する場合、次のデータベーススクリプトを実行してこのサンプル内で使用されているテーブルとデータを作成することもできます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\CreateNullableFields.sql**.

簡素化のために、データベーステーブルは以下のように説明されています。両方のテーブルは、個人のデータを保管しており、同じ列が存在しています。 更に、列 email は、両方のテーブル内のnull データを含むことができます。

```
+----+-----------+-----------+-------------------------+
```

```
| id | firstname | lastname  | email                     |
+----+-----------+-----------+---------------------------+
| 1  | Toby      | Hughey    | t.hughey@nanonull.com     |
| 2  | Mia       | Dahill    | NULL                      |
| 3  | Fred      | Weinstein | f.weinstein@nanonull.com  |
+----+-----------+-----------+---------------------------+
```

SOURCE テーブル

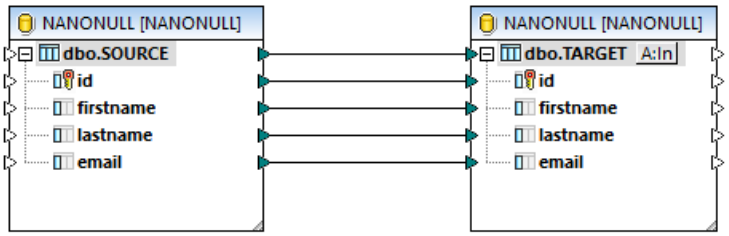
```
+----+-----------+-----------+---------------------------+
| id | firstname | lastname  | email                     |
+----+-----------+-----------+---------------------------+
| 1  | Mia       | Dahill    | NULL                      |
| 2  | Fred      | Weinstein | f.weinstein@nanonull.com  |
+----+-----------+-----------+---------------------------+
```

ターゲットテーブル

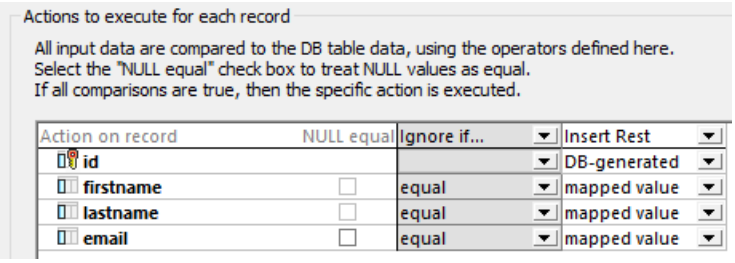
SOURCE テーブルからのデータをTARGET テーブルにマージすることが目的と過程します。新規のレコードのみがTARGET テーブルに挿入されます（このサンプルでは、「Robie Hughey」）。両方のテーブルに存在するレコード（「Mia Dahill」と「Fred Weinstein」）は、無視されます。

タスクは以下のように達成することができます。

1. **「挿入」**メニューから **「データベース」**を選択します。ウィザードステップに従い、データベースに接続します。（次を参照：データベースに接続）。データベースオブジェクトを追加するようにプロンプトされると、テーブルSOURCE を選択します。
2. **「挿入」**メニューから **「データベース」**を選択します。データベースに再接続し、テーブルTARGET をマッピングに追加します。
3. ソースとターゲットコンポーネント間のマッピング接続を描きます。



4. **Action:Insert** A:In ボタンをクリックして、以下のようにデータベーステーブルアクションを構成します：



上記のとおり、「Ignore if.. Insert Rest」アクションの組み合わせが定義されます。この構成は、各レコードに対してマッピングが以下をチェックします：

- ソース内のfirstname は、ターゲット内のfirstname と等しいです。そして、
- ソース内のlastname は、ターゲット内のlastname と等しいです。そして

- ソース内のemail は、ターゲット内のemail と等しいです。

上のすべての条件が満たされる場合、条件に応じて)レコードは無視されます。それ以外の場合、新規のレコードは、ターゲットテーブルに挿入されます。新規のレコードのid は、データベースにより生成されますが、他のフィールド(firstname、lastname、email) は、ソースからマップされた値を使用して表示されます。

email の横のチェックボックスは、このフィールドのための NULL-対応の比較を有効化、または、無効化することができます。email はNull の値を含むことができるため、このチェックボックスを選択する必要があります。(すなわち「Mia Dahill」にはNULL 電子メールアドレスが存在するからです)。チェックボックスの役割を確認するには、データベースを2度更新してください:最初はチェックボックスを選択し更新し、2度目はチェックボックスのチェックが解除された状態で更新してください。

データベースを更新するには、**出力** タブをクリックして、メニューコマンド **出力 | SQL-スクリプトの実行** を実行します。

チェックボックスが選択されている場合、MapForce はNULL フィールドを等価として扱うように明示的に指定されています。ですから、レコード「Mia Dahill」はターゲットテーブルに挿入されません。これは期待されている結果です。

チェックボックスが選択されていない場合、希望する結果ではないレコード「Mia Dahill」が(このレコードは既存にもかかわらず)ターゲットテーブルに挿入されます。MapForce にNULL 値を等価として扱うように明示的な指定が与えられていないからです。データベースに次のクエリを実行すると同様のシチュエーションが発生する可能性があります(NULL 値が「=」演算子と比較され、NULL 対応ではないためこのクエリは、レコードを抽出しません):

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND
lastname = 'Dahill' AND email = NULL;
```

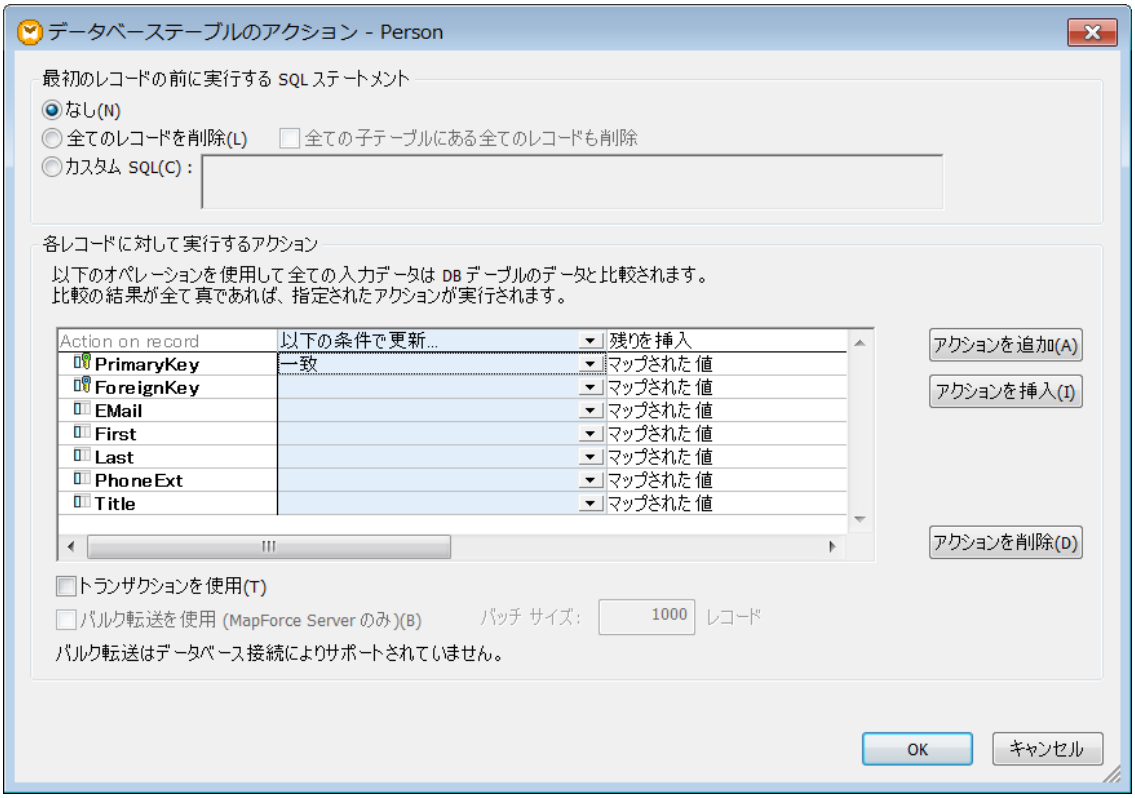
NULL に対応するには、上のクエリ以下のように書かれる必要があります:

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND
lastname = 'Dahill' AND email IS NULL;
```

メモ: 上のクエリは、説明のためで、MapForce により生成される内部クエリの実際の構文を反映していません。(異なるデータベースベンダーはNULL 比較を異なるアプローチで処理するため)NULL 対応が有効化されると、MapForce は、データベース型に応じて生成されたクエリの構文を適応します

7.2.4.12 データベーステーブル アクション設定

データベーステーブルにマッピング接続を作成する都度、データベースアクションボタンが影響を受けるテーブルの横に表示されます。このボタンをクリックすると、データベースの挿入、更新、アクションの削除、とその他のオプションを構成することのできる、データベース テーブル アクション ダイアログボックスが開かれます。



データベース テーブル アクション ダイアログボックス

以下は、データベース テーブル アクション ダイアログボックスで使用することのできる設定の説明です。

最初のレコードの前に実行する SQL ステートメント

このグループボックスでは、最初のレコードがデータベースから取得される前に実行する SQL ステートメントを指定することができます。以下にあるラジオボタンから選択を行ってください：

- **無し** (デフォルトの設定)：アクションは取られることはありません。
- **全てのレコードを削除：** 「各レコードに対して実行するアクション」グループボックスにて定義されたテーブルアクションを実行する前に、選択されたテーブルにある全てのレコードを削除します。**全ての子テーブルにある全てのレコードも削除**チェックボックスを有効にすることで、選択されたテーブルにある子テーブル内にあるデータを取り除くことができます。例えば、次を参照してください[データを複数にリンクされているテーブルに挿入する]
- **カスタム SQL**：テーブル全体を対象としたカスタム SQL ステートメントを記述することができます。

各レコードのために実行するアクション

設定は、マッピングが実行されるときにこのテーブルに対して取られるデータベースアクションを指定します。テーブルアクションを管理するには、**「アクションの追加」**をクリックし、**「アクションの挿入」**または**「アクションの削除」**ボタン.複数のアクションを必要であれば定義することができます (例えば、「残りを挿入」アクションが後に続く「以下の条件で更新 ...」アクション。

定義されたテーブルアクションは、左から右に処理されます。上の例では、「以下の条件で更新 ...」アクションが最初に処理されます。更新の条件 が満たされない場合、次のアクションが処理されます (この例では、「残りを挿入」アクション)。以下

の点に留意してください:

- 1つのアクションのすべての定義された条件が実行されるテーブルアクションのために満たされなければなりません。条件が全て満たされる場合、これらすべてのフィールドは、マッピング上でソースとターゲットアイテムの間の接続が存在する場所で更新されます。 (条件が一致するアクションの右にある)その後のテーブルアクションは、そのレコードのために無視されます。
- 定義された条件が満たされない場合、テーブルアクションはスキップされ、次のアクションが右に向かって処理されます。
- 条件が満たされない場合、テーブルアクションは発生しません。

> 列の条件が入力アクションのために存在しないため、「全て挿入」または「残りを挿入」アクションの後に定義されているアクションは、実行されません。この場合、その後のテーブルアクション列が削除されることを通知するダイアログボックスが表示されます。

「NULL Equal」列内のレコードの横にあるチェックボックスにより、適用できる場合は、MapForce に明示的に、全ての列がNULL 値を持ちそのように扱われることを命令することができます (次を参照してください: データベーステーブルアクション内のNull の扱い)。

他のテーブルへの外部キーリレーションシップを持つマッピングを更新する場合、次のオプションを使用することができます:

子テーブル内のデータの削除	親テーブルのために「以下の条件で更新 ...」アクションを選択すると、このオプションは、役に立ちます。を選択すると、ソースファイル内のレコードの数がターゲットデータベース内のレコードの数と異なる場合、必要になる可能性があり、データベースを同期化 (子テーブル内で孤立下データがない状態に)しておくとよいかもしれません。以下も参照してください: 親テーブルを更新する際の子テーブルのためのオプション.
入力子データを無視する	テーブルの子テーブルレコードに影響を及ぼすこと無く、ターゲット親テーブルを更新するためにこのオプションを使用することができます。以下も参照してください: 親テーブルを更新する際の子テーブルのためのオプション.

アクションの多種の組み合わせを説明している例に関しては、以下を参照してください:

- データをテーブルに挿入する
- データを複数にリンクされているテーブルに挿入する
- テーブルの更新
- 親テーブルを更新する際の子テーブルのためのオプション
- 「以下の条件で更新し、残りを挿入」アクション
- 「以下の条件で削除 ...」アクション
- 「以下の条件で無視 ...」アクション

トランザクションの使用

この特定のテーブルアクションのためのデータベーストランザクションを有効化します。詳細に関しては、次を参照してください: トランザクションロールバックの使用。

バルク転送の使用

(複数のINSERT ステートメントを1つのクエリとしての)バルク転送を有効化します。マッピングがMapForce Server により実行されている場合およびデータベースアクションが「全て挿入」の場合、バルク転送はサポートされています。詳細に関しては、バルク挿入 (MapForce Server) を参照してください。

7.2.5 データベースのデータのジョイン

データベースからデータを読み取るマッピング内では、データベースオブジェクト（テーブルまたはビューなど）を、マッピングにジョインコンポーネントを追加することによりジョインすることができます。例えば、データがリレーショナルデータベース内に保管される外部キーリレーションシップによりバウンドされた 2つ、または、複数のテーブルからのデータを組み合わせることができます。INNER JOIN オペレーションを使用して2つまたは複数のテーブルがジョインされた箇所にデータベースSQL クエリを実行する結果と同じ結果になります。

ジョインコンポーネントに接続されているデータの型により、ジョインの操作は、標準（非 SQL）モード、または、SQL モード内で行うことができます。非 SQL モード内のジョインは、マッピングエンジンにより実行され、SQL モード内のジョインは、マッピングがデータを読み取るデータベースにより実行されます。

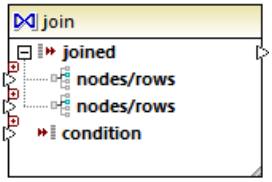
非 SQL モード内のジョインは、コンポーネント型を入力としてサポートするため柔軟性に富んでいます（例えば、ジョインは異なるデータベースからのテーブル間、または、XML 構造とデータベーステーブル間であることができます）。非 SQL ジョインの例は、次を参照してください：サンプル：XML 構造のジョイン。その一方、非 SQL ジョインの実行は、マッピングエンジンにとり更にメモリを必要とする実行となります。（これは、比較の総数がジョインされた構造のクロスジョイン、または、デカルト積を表すためです）。通常、この過程は、データを中心としたマッピング内では、この過程はとても速く影響が小さいです。しかしながら、ジョインしたデータソースが大量の記録により構成される場合、マッピングは実行するために時間を要します。

SQL モード内のジョインは、使用することのできる（テーブルまたはビューなどの）データベースオブジェクトを入力として受け入れます。しかしながら、データベースによりネイティブにより良いマッピングを行うため、柔軟性のある非 SQL ジョインとしては受け入れません。詳細に関しては、次を参照してください：SQL モード内のジョインについて。

メモ： ジョインコンポーネントの使用のみがデータベーステーブル、または、ビューをジョインする唯一の方法ではありません。データベースへのジョインの適用はSQL SELECT ステートメントを使用して行われます データベースデータの選択 (SQL SELECT) を参照してください。SQL SELECT ステートメントのジョインコンポーネント主要な違いは、前者は手動で作成され、そのために柔軟性があります。しかしながら、SQL ステートメントを手動で作成を希望しない場合、ジョインコンポーネントは簡単な代替方法です。

ジョインコンポーネントを追加する

1. マッピング変換言語をBUILT-IN に設定します（これを行うには BUILT IN ツールバーボタンをクリックします、または、**出力 | Built-In 実行エンジン**」メニューコマンドを使用します）。
2. **挿入**」メニューから**『ジョイン』**をクリックします。または、**『ジョイン』**（ ⋈ ）ツールバーボタンをクリックします。ジョインコンポーネントがマッピングに表示されます。デフォルトでは2つの構造からデータを受け入れるため、2つの `nodes/rows` 入力が存在します。必要であれば、**『入力の追加』**（ ⊙ ）ボタンをクリックして、ジョインに新しい入力を追加することができます。以下を参照してください3つ以上の構造をジョインする。



3. ジョインコンポーネントの`nodes/rows` アイテムとジョインされる構造を選択します。
4. ジョイン（または複数の条件）のための条件を追加してください。これを行うには、ジョインコンポーネントを右クリックし、**プロパティ**を選択します。ジョインの条件は、関数のブール値の結果をジョインコンポーネントの`condition` アイテムに接続することにより、マッピングから直接追加することができます。データベースがジョインされる特定の場合、ジョインの条件は自動的にMapForce により作成されます。詳細に関しては、次を参照してください：ジョインの条件の追加。

メモ:

- ジョインコンポーネントは、BUILT-IN 実行エンジン内でのみサポートされています。C#、C++、または Java 内でのコードの生成はサポートされていません。
- 構造がサポートされていない、または、ジョインのための入力ソースにより サポートされて以内場合、 MapForce マッピング上直接ヒントを表示するか、または、マッピングを検証する際に、メッセージウィンドウ内にヒントを表示します。マッピング (次を参照してください:マッピングの検証).
- ジョインコンポーネントをインラインユーザー定義関数の結果または入力パラメーターに接続しないでください。このような接続が存在すると、マッピングの検証中に検証エラーが発生します。
- 使用することのできるデータベースコンポーネント (テーブルまたはビューなどの)を直接ジョインコンポーネントに接続すると、**SQL モード**(SQL) ボタンがジョインコンポーネントの右上に自動的に表示されます。有効化されると、このボタンは、ジョイン操作に適用することのできる特別な SQL 機能を提供します (次を参照してください:SQL モード内のジョインについて)。
- 3つ以上の構造をジョインする必要がある場合、複数のジョインコンポーネントを追加して、 重ねて」表示します (すなわち、ジョインコンポーネントの出力を他のジョインコンポーネントの入力として与えます)。ジョインされたシーケンス全部を他のジョインの入力としては使用できないことに注意してください。ジョインされた アイテムの出力を他のジョインコンポーネントに接続しようと試みると、エラーメッセージが表示されます。しかしながら、ジョインの結果の一部を他のジョインに接続することは可能です。
- SQL モードでは、ジョインを重ねて表示することはサポートされていません。しかしながら、ジョインされた子孫 (子) テーブルを他のジョインに接続することは可能です。代わりに、SQL SELECT ステートメントを使用することができます (次を参照 :仮想テーブルとしての SQL SELECT ステートメント)。親と子テーブルに関する詳細は、次を参照してください:データベースリレーションシップの扱い方。

7.2.5.1 SQL モード内のジョインについて

使用することのできるデータベースコンポーネント (テーブルまたはビューなどの)を直接ジョインコンポーネントに接続すると、**SQL モード**(SQL) ボタンがジョインコンポーネントの右上に自動的に表示されます。SQL モードが有効化されていると ジョインオペレーションは、マッピングがデータを読み込む箇所でデータベースにより行われます。すなわち、MapForce は、ジョインに使用される全てのテーブルからのデータを選択し結合するために適切な SQL 構文を使用して、内部でデータベースにクエリを送信します。重要な点は、SQL を書く必要がない点です。ジョインコンポーネントがマッピング上でデザインされた方法をベースにして必要なく構が生成されます。次のサンプルを参照してください。

メモ: データベースとSQL パースペクティブから、MapForce により生成されたジョインは常にINNER ジョインです。これは、両方の入力セット内の条件を満たすレコードのみがジョインコンポーネントにより返されます。

SQL モードを可能にするためには、以下の条件が満たされている必要があります:

1. ジョインされる両方のオブジェクト (テーブルまたはビュー)は、同じデータベースから与えられなければなりません。
2. ジョインされる両方のオブジェクトは、同じMapForce コンポーネントから与えなければなりません。 (コンポーネント内でデータベースオブジェクトを追加および削除する方法は以下のとおりです:データベース コンポーネントを右クリックして、コンテキストメニューから**追加 削除 編集 データベースオブジェクト**を選択します)。
3. ジョインの条件 (または条件)は、マッピング上ではなく、コンポーネント プロパティからのみ定義されます (ジョインコンポーネントのヘッダーを右クリックして、**プロパティ**を選択します) (次を参照してください:ジョインの条件の追加)。

メモ: SQL モード内でデータベーステーブルがジョインされると、MapForce は、テーブル間で検出された外部キーリレーションシップをベースにして、ジョインの条件を自動的に作成します。自動的なジョインの条件が起こるには、データベーステーブルがMapForce コンポーネント上で親子関係内に存在する必要があります (すなわち、1つのテーブルが 親」または 子」で、他のテーブルがコンポーネント上に存在する必要ああります)。サンプル:SQL モード内のテーブルのジョインを参照してください。

4. すべてのデータベーステーブルは、現在のターゲットコンテキスト内にまだ存在することはできません。例えば、ジョインの操作前にマッピングがテーブルをクエリするようにデザインされている場合、ジョインを行うことは不可能になります。マッピングの実行に関する詳細は、次を参照してください:マッピングのルールと戦略。

ジョインコンポーネントの右上にある**SQL** (SQL) ボタンを使用して、SQL モードを閲覧または管理することができます :

SQL SQL モードが無効化されています。

SQL SQL モードが有効化されています。

SQL ボタンが不足している場合、SQL モードは意味を成さないことを示します 、または、ジョインされたデータのためにサポートされていないことを意味します。

特定のケースの場合、SQL モードは明示的に無効化されている必要があります (SQL)、例 :

- 異なるデータベースからのテーブルをジョインする場合。異なるデータベース言語間の差異のため異なるデータベース間のSQL モードでのジョインは不可能です。しかしながら、標準 (非 SQL) には制約は適用されません。
- ジョインコンポーネント プロパティの外部でマッピングがジョインの条件を必要とする場合、 (条件がマッピング上で定義されており、ジョインコンポーネントの`condition` アイテムに接続されている場合)。

ジョインされたデータベーステーブル、または、ビューに同一のフィールド名が含まれていることが頻繁にあります。SQL モードが有効化されると、このようなアイテムは、キーワード「AS」によりプレフィックスされたコンポーネント上に表示されます。例えば、2つのジョインされたテーブルが「id」フィールドを含む場合、このフィールドは、最初にジョインされたテーブル上で「id」として表示されます、そして、2番目にジョインされたテーブルでは「id AS id2」として表示されます。ジョインされたテーブルは、例えば、同じテーブルがテーブル内でジョインされた場合、エイリアス名を生成することができます。

マッピング上で後に参照する場合エイリアス フィールドまたはテーブル名は、重要です。例えば、ジョインコンポーネントの出力がSQL WHERE/ORDER コンポーネントに接続されている場合などです。目的を達成するには、ジョインコンポーネントの出力を、SQL WHERE とORDER BY 句を入力するSQL WHERE/ORDER コンポーネントに接続することができます。

WHERE 句からフィールドを参照するには、フィールド名が後に続き、ピリオド (.)文字が後に続く テーブル名を書き込みます。テーブルエイリアスを参照するには、ジョインコンポーネントに表示される順のエイリアス名を使用します。ORDER BY 句内では、同じテクニック (`table.field`) 、または、(「AS」の後に表示される名前である)エイリアスフィールド名を書き込みます。

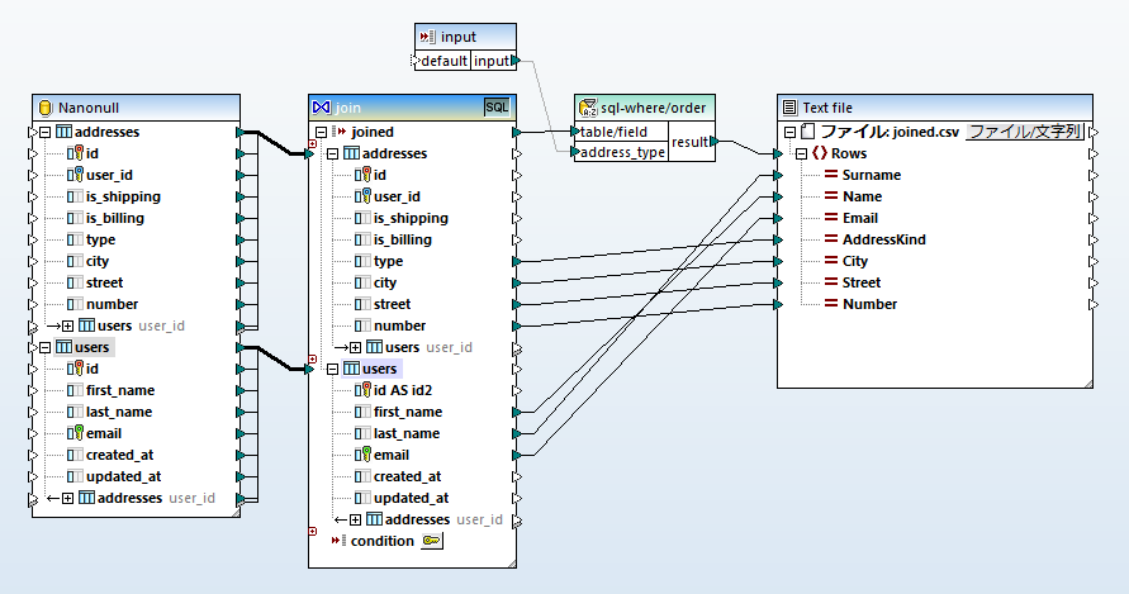
SQL WHERE/ORDER 句を使用するマッピングのサンプルに関しては、サンプル :SQL モード内のジョインテーブル を参照してください。 .

データベーステーブルとジョインコンポーネント間は許可されていません。ジョインコンポーネントの 前ではなく後に追加することができます。SQL WHERE/ORDER コンポーネントの詳細に関しては、次を参照してください :データベースデータのフィルターと並べ替え (SQL WHERE/ORDER)。

7.2.5.2 サンプル SQL モード内でテーブルをジョインする

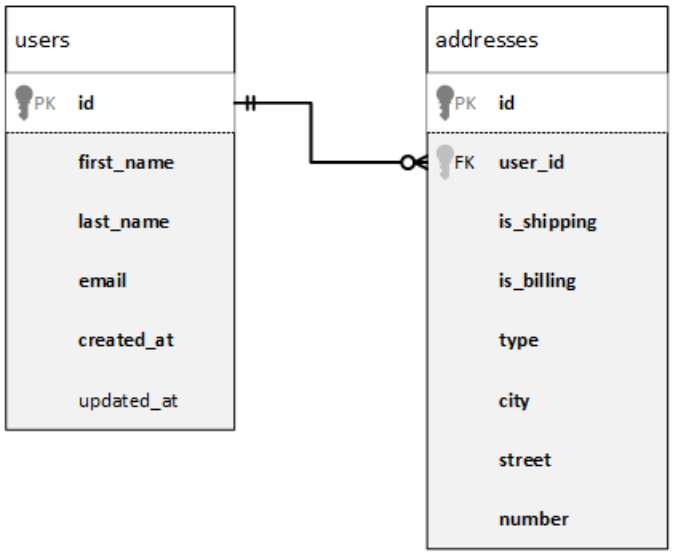
このサンプルは、MapForce ジョイン コンポーネントを使用した、データベーステーブルからのデータをジョインする方法について説明しています。ジョインの操作は、SQL モード内のジョインについてで説明されているように、SQL モード内で実行されます。3つ、または、以上のテーブルワークを同様にジョインする場合も同様に行います。以下も参照してください :3つ以上の構造をジョインする。

このサンプルは、次のパスにあるマッピングサンプルが存在します :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\JoinDatabaseTables.mfd**。

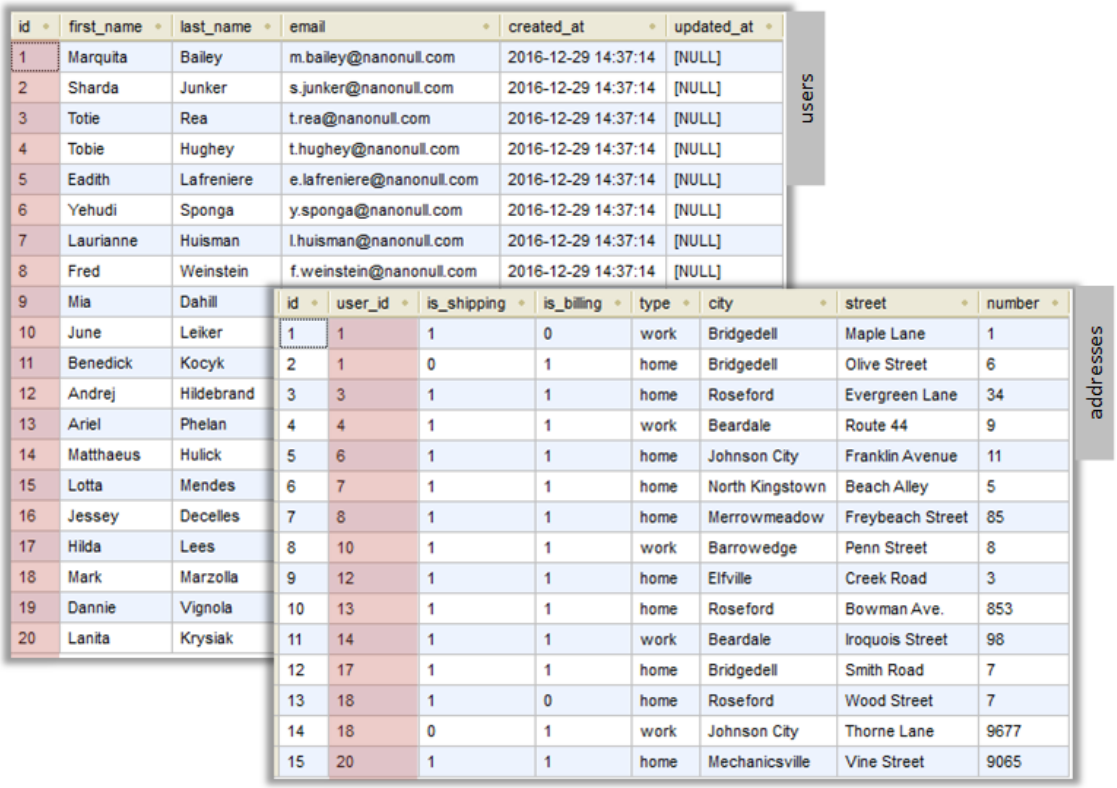


JoinDatabaseTables.mfd

上のマッピングの目的は、2つのソースデータベーステーブルからのデータを単一のターゲットCSV ファイルに結合することです。下のデータベースダイアグラムに示されているとおり、最初のテーブル (users) は個人の住所を保管し、2番目のテーブル (addresses) は個人の名前と電子メールアドレスを保管します。2つのテーブルは、共有のフィールドによりリンクされています (users 内の id が addresses 内の user_id に対応しています)。データベース用語では、この種類のリレーションは「外部キーリレーションシップ」と呼ばれています。



簡素化のために、下のイメージは両方のテーブルの実際のデータを表示しています。



users

id	first_name	last_name	email	created_at	updated_at
1	Marquita	Bailey	m.bailey@nanonull.com	2016-12-29 14:37:14	[NULL]
2	Sharda	Junker	s.junker@nanonull.com	2016-12-29 14:37:14	[NULL]
3	Totie	Rea	t.rea@nanonull.com	2016-12-29 14:37:14	[NULL]
4	Tobie	Hughey	t.hughey@nanonull.com	2016-12-29 14:37:14	[NULL]
5	Eadith	Lafreniere	e.lafreniere@nanonull.com	2016-12-29 14:37:14	[NULL]
6	Yehudi	Sponga	y.sponga@nanonull.com	2016-12-29 14:37:14	[NULL]
7	Laurianne	Huisman	l.huisman@nanonull.com	2016-12-29 14:37:14	[NULL]
8	Fred	Weinstein	f.weinstein@nanonull.com	2016-12-29 14:37:14	[NULL]
9	Mia	Dahill			
10	June	Leiker			
11	Benedick	Kocyk			
12	Andrej	Hildebrand			
13	Ariel	Phelan			
14	Matthaeus	Hulick			
15	Lotta	Mendes			
16	Jessey	Decelles			
17	Hilda	Lees			
18	Mark	Marzolla			
19	Dannie	Vignola			
20	Lanita	Krysiak			

addresses

id	user_id	is_shipping	is_billing	type	city	street	number
1	1	1	0	work	Bridgedell	Maple Lane	1
2	1	0	1	home	Bridgedell	Olive Street	6
3	3	1	1	home	Roseford	Evergreen Lane	34
4	4	1	1	work	Beardale	Route 44	9
5	6	1	1	home	Johnson City	Franklin Avenue	11
6	7	1	1	home	North Kingstown	Beach Alley	5
7	8	1	1	home	Merrowmeadow	Freybeach Street	85
8	10	1	1	work	Barrowedge	Penn Street	8
9	12	1	1	home	Elfville	Creek Road	3
10	13	1	1	home	Roseford	Bowman Ave.	853
11	14	1	1	work	Beardale	Iroquois Street	98
12	17	1	1	home	Bridgedell	Smith Road	7
13	18	1	0	home	Roseford	Wood Street	7
14	18	0	1	work	Johnson City	Thorne Lane	9677
15	20	1	1	home	Mechanicsville	Vine Street	9065

`users` テーブル内の各ユーザーはゼロの値、または、`addresses` テーブル内に複数の住所を有することができます。例えば、ユーザーは、「home」の種類に属する1つの住所、または、2つの住所（「home」と「work」の2種類の住所）、または、住所なしを選ぶことができます。

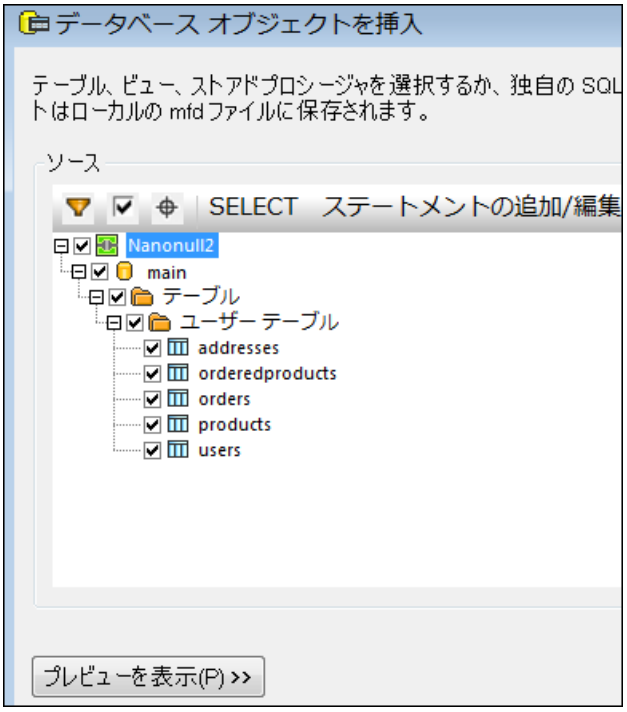
マッピングの目的は住所テーブル内に住所が少なくとも1つ記入されている全てのユーザーのデータ（名、姓、電子メール、住所など）を抽出することです。また、特定の種類の住所を簡単に抽出することもできます（例えば、自宅の住所、または、勤務先の住所のみを抽出することができます）。（「home」、または、「work」）などの抽出する住所は、マッピングにパラメーターとして与えられます。抽出された個人のレコードは姓によりアルファベット順に並べ替えられます。

下で説明されているとおり、ジョインコンポーネントを使用して、マッピングの必要条件が満たされます。

メモ: ジョインコンポーネントの使用のみがデータベーステーブル、または、ビューをジョインする唯一の方法ではありません。データベースへのジョインの適用はSQL SELECT ステートメントを使用して行われます（データベースデータの選択 (SQL SELECT) を参照してください）。SQL SELECT ステートメントのジョインコンポーネント主要な違いは、前者は手動で作成され、そのために柔軟性があります。しかしながら、SQL ステートメントを手動で作成を希望しない場合、ジョインコンポーネントは簡単な代替方法です。

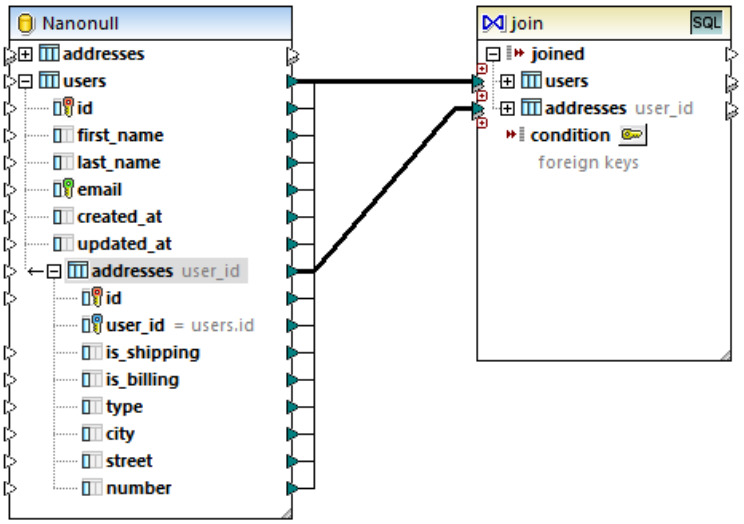
ステップ 1：ソースデータベースを追加する

1. **「挿入」**メニューから **「データベース」**を選択します。（または、**データベースの挿入** ツールバーボタン）をクリックします。
2. 「SQLite」をデータベースの型として選択し、**「次へ」**をクリックします。
3. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にある **Nanonull.sqlite** ファイルを参照します。
4. プロンプトされると、`addresses` と`users` テーブルを選択します。



ステップ 2：ジョインコンポーネントを追加する

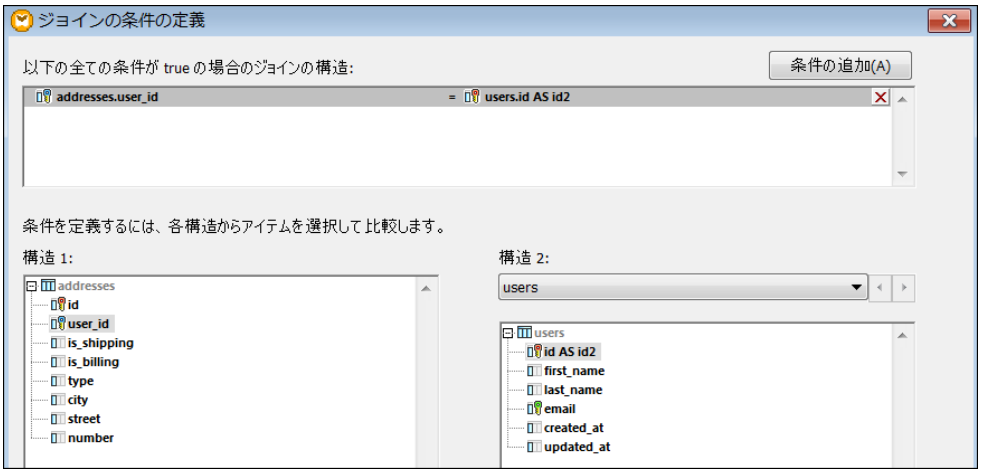
1. **挿入** メニューから **ジョイン**（または、**ジョイン** ツールバーボタン）をクリックします。
2. `users` テーブルから、ジョインコンポーネントの入力に接続を描きます。
3. `users` テーブルを展開し、`addresses` テーブル（`users` の子）からジョインコンポーネントの2番目の入力に接続を描きます。 ボタンにより必要に応じて更にテーブルを追加することができます。しかしながら、このサンプルでは、2つのテーブルのみがジョインされます。



メモ： `addresses` テーブル（`users` の子ではない）から直接接続を追加することができます。しかしながら、この場合、ジョイン条件の追加で説明されているとおり、ジョインの条件は手動で定義されなければなりません。このサンプルの目的のために、上に示されるように接続を作成してください。これにより、必要なジョインの条件が自動的に作成され

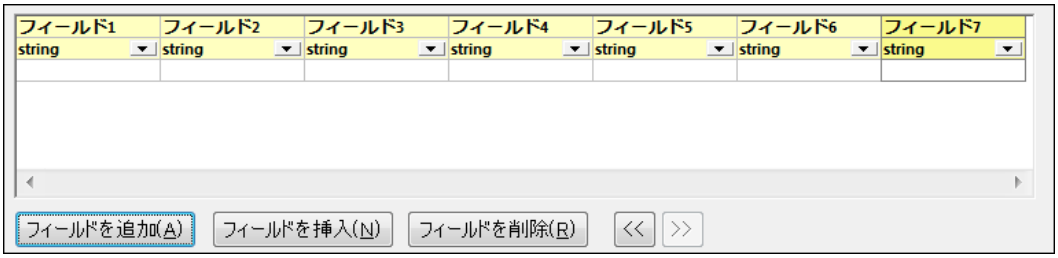
ます。

4. ジョインコンポーネント上で使用することのできる **「ジョインの条件の定義」** ボタンをクリックします。ジョインの条件が自動的に作成される事に注意してください。 (`users.id = addresses.user_id`)。

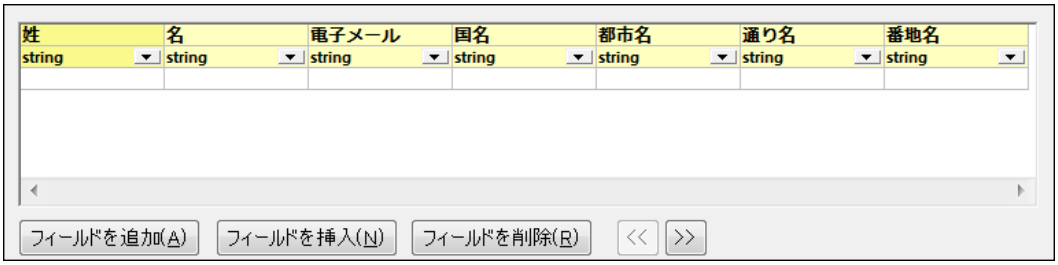


ステップ 3 : ターゲット CSV コンポーネントの追加

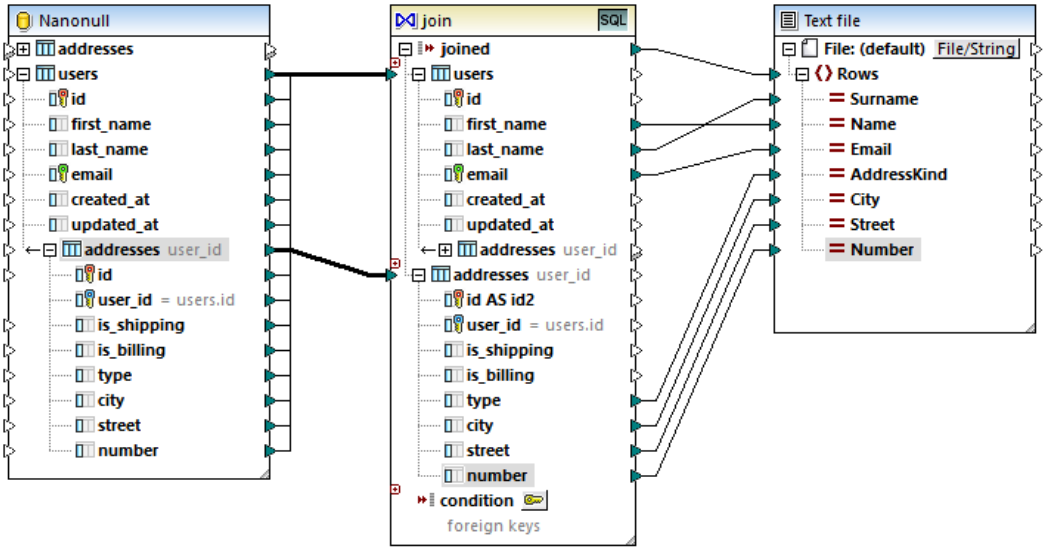
1. **「挿入」** メニューから **「テキストファイル」** (または、**テキストファイルの挿入** ツールバーボタン) をクリックします。 .
2. テキストの処理モードを選択されるように問われると、**標準 CSV のために簡単処理を使用する ...** を選択します。
3. **「フィールドの追加」** を数回クリックして、7 つの CSV フィールドを作成します。他の設定を保持します。



4. 各フィールドのタイトルのセルをダブルクリックして、詳細を説明する名前を与えます (これによりマッピングを簡単に読み込むことができます)。

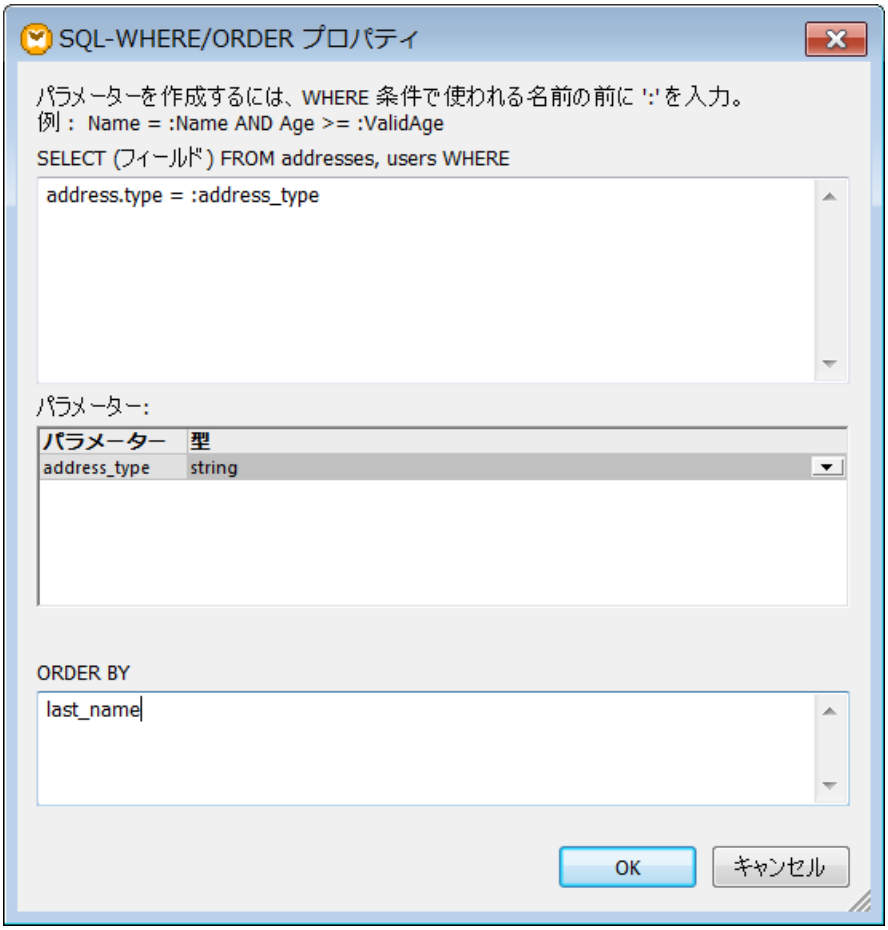


5. ジョインコンポーネントと CSV コンポーネント間の接続を下に示されるように描きます。ジョインコンポーネントの `joined` アイテムとターゲットコンポーネントの `Rows` アイテム間の接続は、ジョインの条件を満たすために、ターゲット内に無制限の数量の (行) を作成することを意味します。

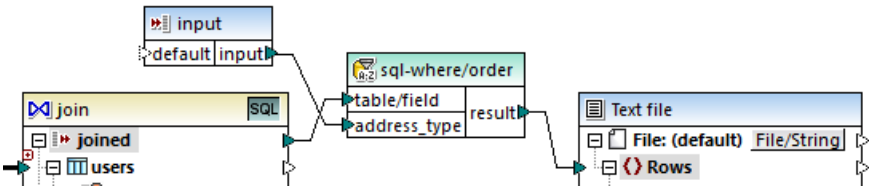


ステップ 4：SQL WHERE/ORDER 条件と入力パラメーターを入力する

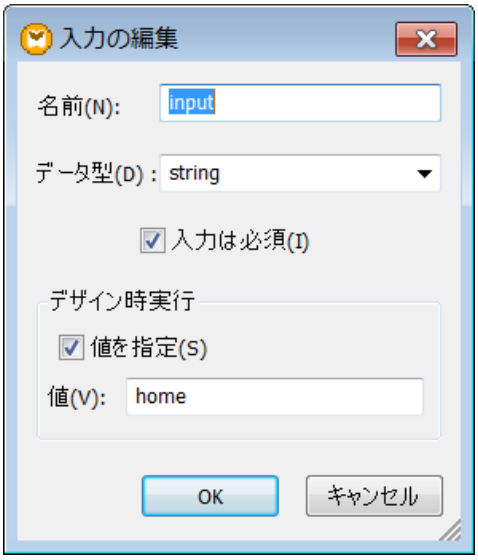
1. ジョインコンポーネントの `joined` アイテムとターゲットコンポーネントの `Rows` アイテム間の接続を右クリックし、**「SQL-WHERE/ORDER を挿入する」**を選択します。
2. WHERE とORDER BY 句を下に示されるように入力します。



3. マッピング上で、(**挿入 | 入力の挿入** メニューコマンドを使用して)入力コンポーネントを追加し、前のステップで作成された`address_type` パラメーターに出力を接続します。



4. 入力コンポーネントをダブルクリックして、下に示されるように構成します。 MapForce 内でマッピング出力をプレビューするためにデザイン時刻 値 (この場合、 「home」)が必要とされます。職場の住所を抽出するためにプレビューを行うには、この値を 「work」と置き換えてください。



マッピングの説明

ステップ 2 は定義されているジョインの条件 `users.id = addresses.user_id` を満たすレコードのみがターゲットにコピーされることを確認します。2つのテーブルは外部キーリレーションシップによりバインドされており、マッピング接続は状況に応じて描かれますジョインの条件は自動的に追加されています。 (テーブルリレーションシップに関する詳細は、以下を参照してください:データベースリレーションの扱い)。このサンプルにはお適用することができませんが、手動でジョインの条件を定義することもできます、次を参照してください:サンプル:複数のテーブルからCSV レポートを作成する

2つのソーステーブルは、同じデータベースおよび同じコンポーネントからであり、このジョインは SQL (SQL) モードを活用することができます。SQL モードが有効化されているため、ジョインの操作は MapForce* ではなく、データベースにより行われます。SELECT ステートメントが代わりに作成され、データベースに送信されます。

ステップ 4 で追加された SQL WHERE/ORDER コンポーネントにより、レコードセットを (自宅、または、勤務先の住所を抽出するなど) フィルターし並べ替えることができます。WHERE 句は、型 `string` のパラメーター **`:address_type`** を作成することに注意してください。パラメーターを使用することにより、マッピングから住所の種類 (自宅、または、勤務先) を与えることができます。SQL WHERE/ORDER の詳細に関しては、次を参照してください:データベースデータのフィルターと並べ替え (SQL WHERE/ORDER)。

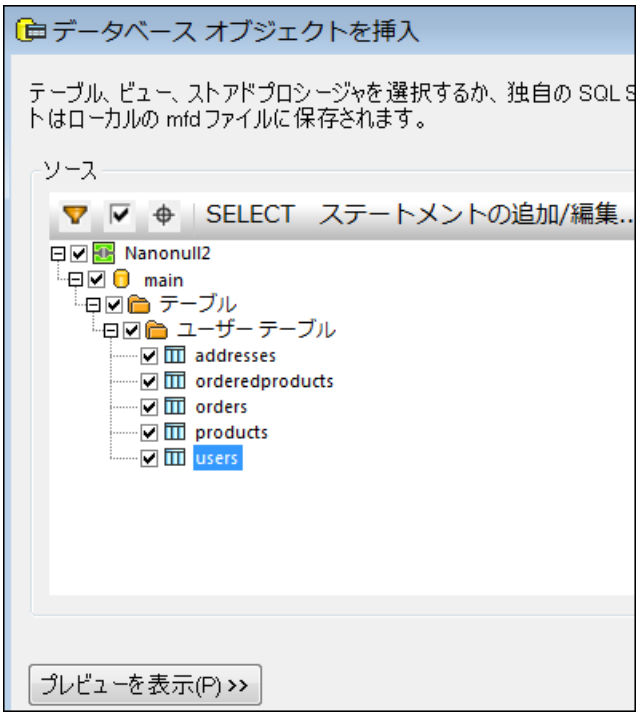
最後に、入力コンポーネントによりマッピングの実行中に実際のパラメーター値を提供することができるようになります。MapForce 外でマッピングが実行される場合、(例えば、異なるマシン上で MapForce Server によりマッピングが実行される場合) 上記のデザイン時刻が無視されるように、コマンドライン パラメーターとしてマッピングのランタイムに入力が与えられる必要があります。詳細に関しては、次を参照してください:パラメーターをマッピングに提供する。

7.2.5.3 サンプル :複数のテーブルから CSV レポートを作成する

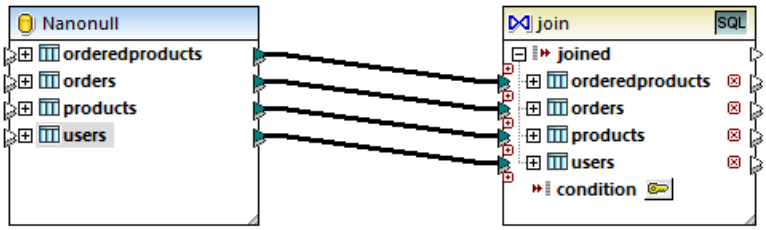
このサンプルは、CSV フォーマットの単一のレポートにデータを抽出する目的のための複数のデータベーステーブルのジョインの方法について説明しています。このサンプルで使用されているデータベースは、**Nanonull.sqlite** で、以下のパスで見つけることができます: **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial**。このデータベースは、(注文、製品、および住所が含まれる) 架空の会社の情報を含んでいます。リレーショナル データベースでは通常、情報が情報が正規化され、複数のテーブルに分割されます。例えば、`users` テーブルには、(姓、名、および電子メールアドレスがふくまれる) ユーザーの個人の情報が保管されます。データベースは、ユーザーにより注文された製品の情報が2つの異なるテーブルに含まれます : (注文の一意の ID 、および注文の時刻を含む) `orders` および (注文された製品のリスト、および、個数が含まれる) `orderedproducts`。更に、製品の名前は `products` という異なるテーブルに保管されます。

このサンプルの目的は、特定の製品を誰が、何個注文したかを明確にするため、異なるテーブルより抽出されるデータをベースにしたレポートの生成です。この目的を達成するには以下の手順を行います：

1. **「挿入」**メニューから**データベース**をクリックします。
2. データベース型を選択するようにプロンプトされると、**「SQLite」**をクリックして、**「次へ」**をクリックします。
3. 上記の**Nanonull.sqlite** データベースを参照し、**「接続」**をクリックします。
4. プロンプトされると、テーブル`orderedproducts`、`orders`、`products` および`users` を選択して、**「OK」**をクリックします。



5. ジョインコンポーネントをマッピングに追加し、4つの`nodes/rows` アイテムを**「入力の追加」**（ ）ボタンをクリックして作成します。
6. データベースからの4つのテーブルをジョインコンポーネントの対応する入力アイテムに接続します。

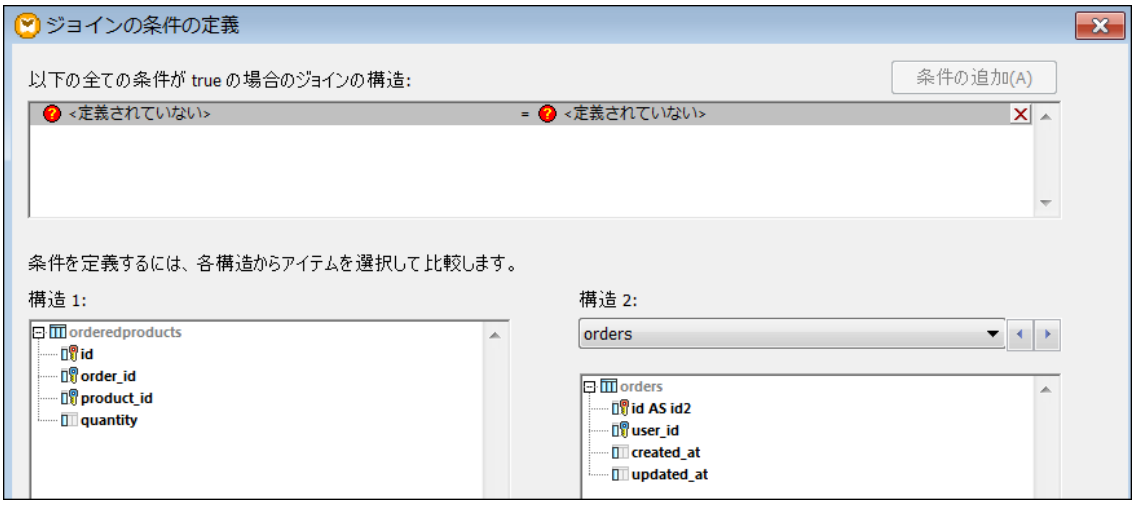


メモ： 別のシナリオでは、全てのジョインされたテーブルが同じ「ルート」テーブルの下でネストされるようにジョインコンポーネントにテーブル`orderedproducts` を接続することができ、また、同じレベルではなく、下のレベルに存在するテーブル`orders` などを接続することができます。以下も参照してください：データベースリレーションシップの扱い。マッピングの結果は、テーブルをジョインした結果と同じになります。差異は、このサンプルでは、ジョインの条件が手動で作成されなければならず、下に示されるように、別のシナリオでは、ジョインの条件がMapForce により自動的に作成されます。ジョインの条件を手動で定義せずにテーブルをジョインするサンプルは、次を参照してください：サンプル：SQL モード内でテーブルをジョインする。同じ「ルート」テーブルの下でジョインされたテーブルの他のマッピングは、以下のパスで検索することができます：**<Documents>\Altova\MapForce2017\MapForceExamples\DB_Denormalize.mfd**。

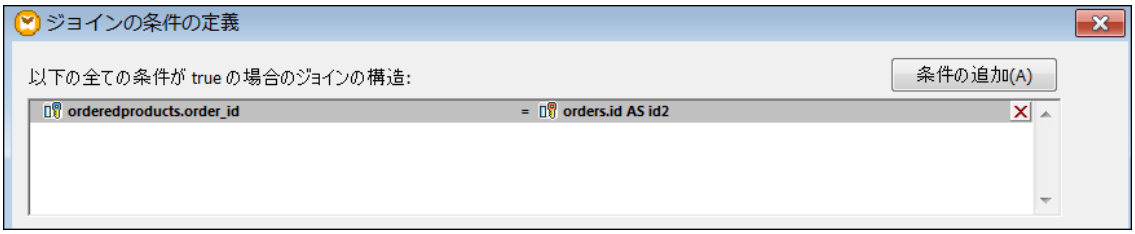
2つ以上の構造（またはデータベーステーブル）がジョインされる場合、ジョインコンポーネント上に表示される順番は重要です。このサンプルではテーブルは、次の順番のジョインコンポーネントに接続されています：

1. orderedproducts
2. orders
3. products
4. users

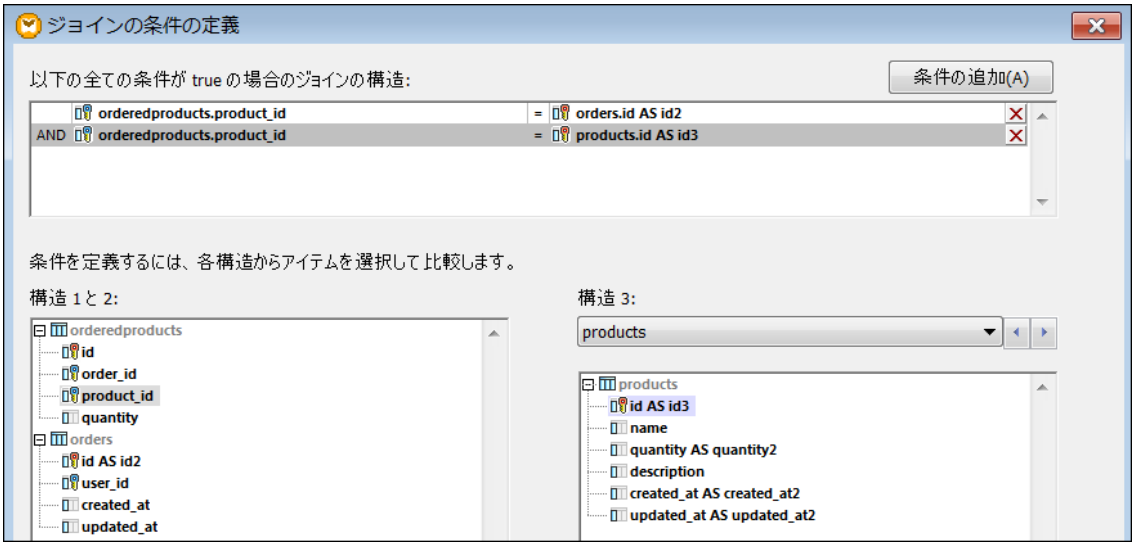
この順序は、**「ジョインの条件の定義」**（ ）ボタンをクリックした時に、どのように対応する構造が「ジョインの条件の定義」ダイアログボックス上で表示されるかに影響を及ぼします。すなわち、最初のテーブル（orderedproducts）はデフォルトで**構造 1**の下に表示され、そのすぐ後のテーブル（orders）は、**構造 2**の下に表示されます。



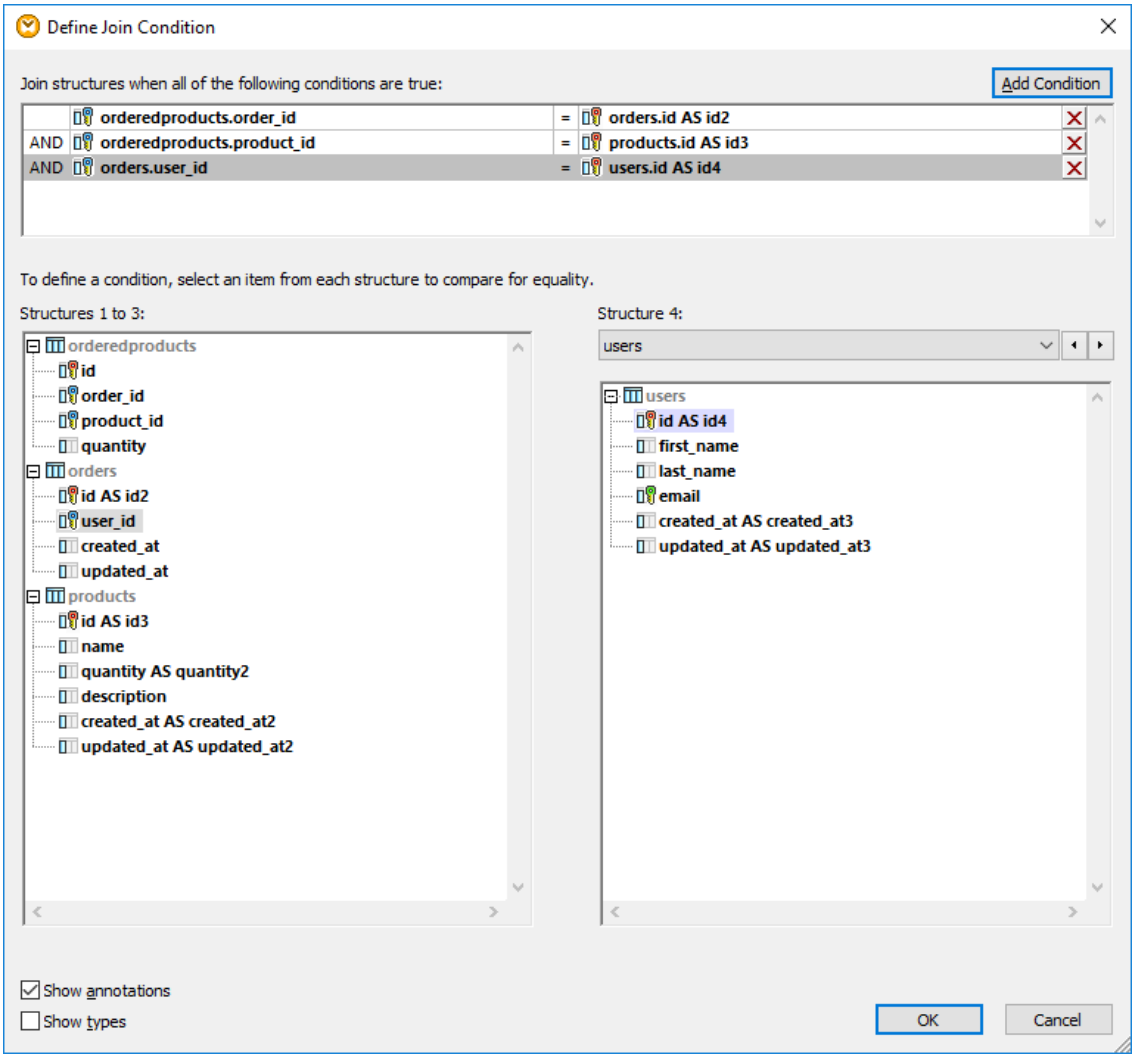
最初のジョインの条件を定義するには、左側のペイン内のorder_id アイテムをクリックし、右側のペイン内のid アイテムをクリックします。orderedproducts.order_id とorders.id. が組み合わされます：



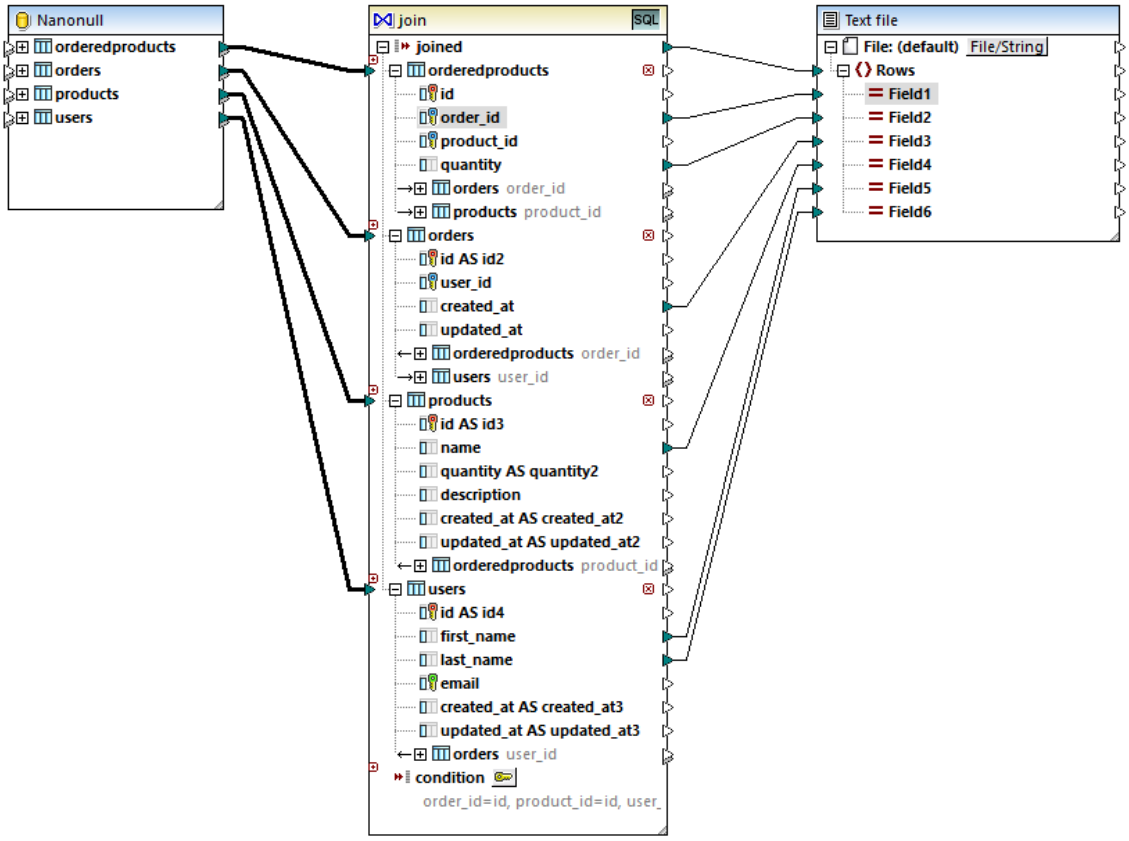
この時点まででは、2つのテーブルがジョインされました。3番目のテーブルを含むジョインの条件を定義するには、右側のペインの上で使用することのできるドロップダウンリストから希望するテーブルを選択します。左側のペインは、この場合、ジョインコンポーネント上でジョインの前に発生する全てのテーブルが表示されています。例えば、（これらのテーブルは、ジョインコンポーネント上でproducts の前に発生するため）products を右側で選択すると、左側は、orderedproducts とorders を表示します。テーブルproducts のフィールドを前のテーブルとジョインすることができます（この場合、orderedproducts.product_id とproducts.id）。



4番目のテーブル (`users`) をジョインするには、`users` テーブルをドロップダウンリストから選択します。フィールド `orders.user_id` と`users.id` を組み合わせることができます。



必要とされるジョインの条件がすべて定義されると、ジョインコンポーネントのアイテムをターゲットコンポーネントにマップすることができます。マッピングを完了するには、CSV コンポーネントを追加し、ジョインコンポーネントからのアイテムを下に示されるようにターゲット CSV コンポーネントに接続します（次を参照してください：CSV とテキストファイル）：



上で説明されているマッピングは、ジョインに含まれる4つのテーブルからコンパイルされた(CSV フォーマットの)レポートを作成します:

- 注文のID (`orderedproducts` テーブルから取得)
- 注文されたアイテムの個数 (`orderedproducts` テーブルから取得)
- 注文の時刻 (`orders` テーブルから取得)
- 注文された製品の名前 (`products` テーブルから取得)
- 製品を注文したユーザーの氏名 (`users` テーブルから取得)

7.2.6 データベースデータのフィルターと並べ替え (SQL WHERE/ ORDER)

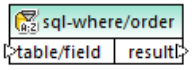
データベースデータをフィルターし、並べ替える場合、SQL WHERE/ORDER コンポーネントを使用します。これにより、データをフィルターするSQL WHERE 句をMapForce グラフィカルユーザーインターフェイスから入力することができます。または、特定のデータベースフィールドによるレコードのセットを昇順および降順に並べ替える場合、ORDER BY 句を指定することもできます。

SQL WHERE/ORDER コンポーネントはデータベースマッピングコンポーネントに接続されている必要があります。ジョインされたセット、または、レコードをフィルターする必要がある場合、SQL WHERE/ORDER をジョインコンポーネントに接続することもできます (次を参照してください: データベースのデータのジョイン)。

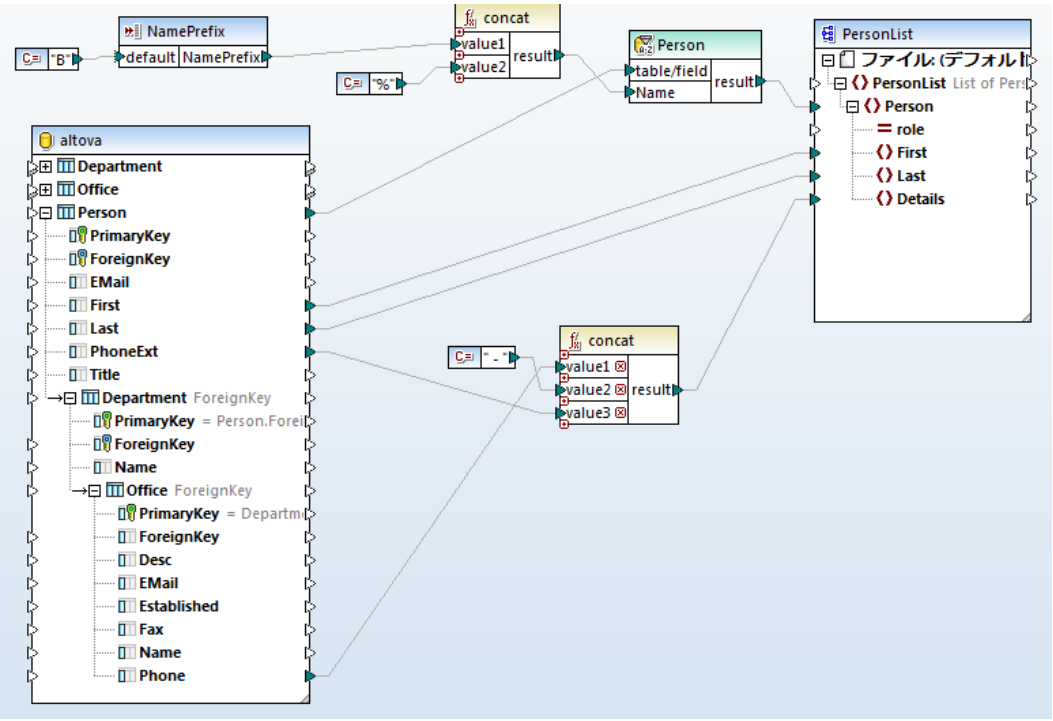
SQL WHERE/ORDER コンポーネントをマッピングに追加する

1. **挿入**」メニューから**SQL WHERE/ORDER** をクリックします。 デフォルトでは、SQL WHERE/ORDER

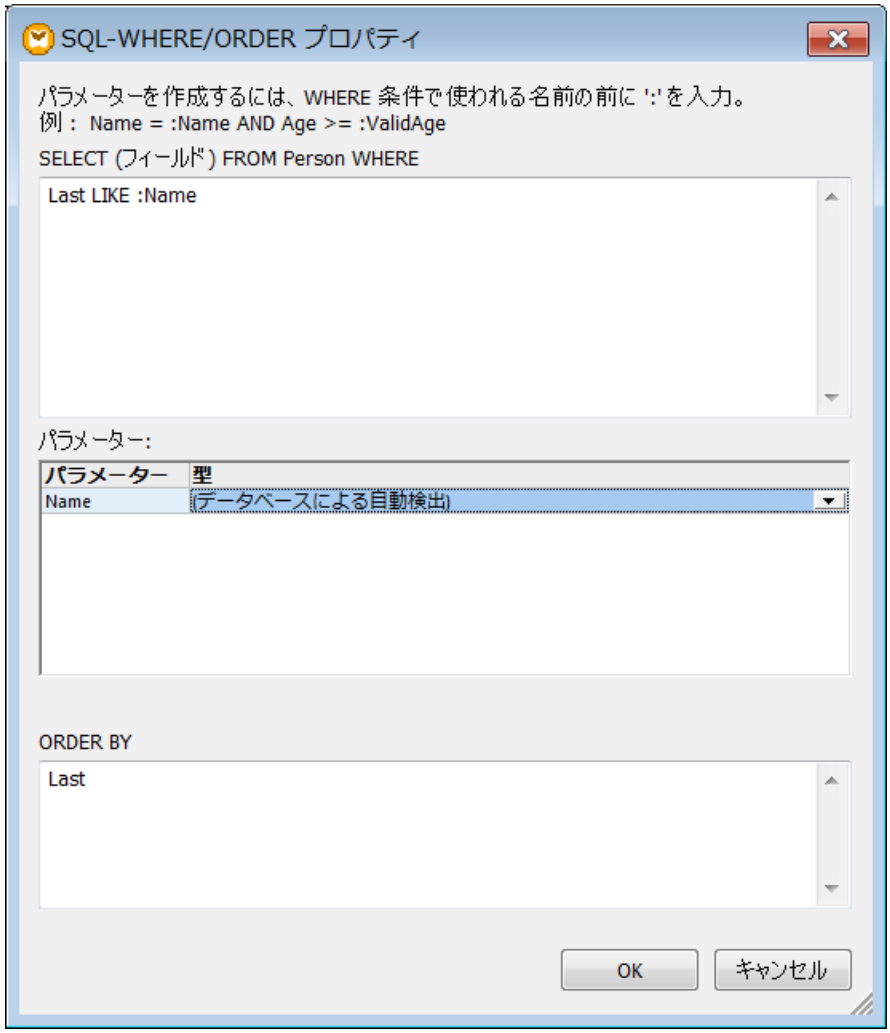
コンポーネントは次の構造を持ちます：



2. ソースデータベーステーブルを接続、または、SQL WHERE/ORDER の `table/field` アイテムのフィールドに接続します。例えば、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダーからマッピング **DB_PhoneList.mfd** を開きます。このマッピングでは、SQL WHERE/ORDER は、ソーステーブル「Person」から、「B」文字から始まる姓を持つ全てのレコードをフィルターします。



3. SQL WHERE/ORDER コンポーネントのヘッダーをダブルクリックします。（または、右クリックしてコンテキストメニューから**プロパティ**を選択します）。これにより、「SQL WHERE/ORDER プロパティ」ダイアログボックスが開かれます。



4. 上のテキストボックスにSQL WHERE 句 を入力します。または、ORDER BY 句を入力します。上のイメージは**DB_PhoneList.mfd** マッピング内で定義されているWHERE とORDER BY 句を示しています (この設定については、下で詳しく説明されています)。例えば、次を参照してください:WHERE とORDER BY 句の作成。

SQL WHERE/ORDER にパラメーターを与える

マッピング**DB_PhoneList.mfd** 内で使用されているSQL WHERE/ORDER コンポーネントは、WHERE 句を次のように定義しています:

```
Last LIKE :Name
```

「Last」は、接続されたテーブル内のデータベースフィールド名を参照しています。「LIKE」は、SQL 演算子です。「Name」は、マッピング上に「Name」という名前のパラメーターを作成します。

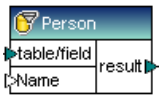
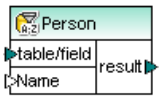
SQL WHERE/ORDER コンポーネント内のパラメーターは任意です。マッピングからWHERE 句に値をパスする際に役に立ちます。パラメーター無しでは、上のWHERE 句は以下のように書くことができます:

```
Last LIKE "B%"
```

これは「B」から始まる姓を持つ全ての個人を抽出します。クエリに更に柔軟性を与えるため、「B%」の代わりにパラメーターを追加します。これによりマッピングから他の文字を与えることができるようになります（例えば、「C」、そして、定数、または、マッピング入力パラメーターを変更するだけで、「C」から始まる姓を持つ個人を抽出できるようになります）。

SQL WHERE/ORDER コンポーネントの表示

SQL WHERE/ORDER コンポーネントの重要な点は設定内の定義により外観が異なる点です。このようにしてSQL WHERE/ORDER コンポーネントの振る舞いを直接マッピングから確認することができます。例：

	WHERE 句 が定義済みです。
	定義済みのパラメーターが存在するWHERE 句。パラメーター「Name」「table/field」アイテムの下に表示されています。
	定義済みのパラメーターが存在するWHERE 句 。更に、ORDER BY 句 が定義済みです。並べ替えは、A-Z 並べ替えアイコンにより表示されています。

SQL WHERE/ORDER ヘッダーにマウスのカーソルをポイントすると、定義されている異なる句を表示するヒントが表示されます。

7.2.6.1 WHERE とORDER BY 句の作成

SQL WHERE/ORDER コンポーネントがマッピングに追加されると、コンポーネントに接続されるデータをフィルターする方法を定義することのできるWHERE 条件（句）が必要になります。WHERE 条件は、MapForce の「SQL WHERE/ORDER プロパティ」ダイアログボックスに入力される必要があります。

MapForce からWHERE 条件を作成することは、MapForce 外で同じSQL 句を作成することと類似しています。対応するデータベースのSQL 言語に適応される構文を使用してください。例えば、演算子、ワイルドカード、サブセレクト、または、集計関数を使用することもできます。マッピングからパスすることのできるパラメーターを作成し、後にパラメーター名が続くセミコロン文字（:）を入力します。

メモ： WHERE 句の作成が終わると、「OK」をクリックします。MapForce は最後のSQL ステートメントの整合性を検証します。構文エラーが存在するとダイアログボックスがプロンプトを表示します。

以下の演算子をWHERE コンポーネントにて使用することができます：

演算子	説明
=	等価
<>	異なる
<	未満
>	を超える
>=	以上
<=	以下

演算子	説明
IN	指定した値をカラムから検索
LIKE	指定したパターンを検索
BETWEEN	指定した範囲を検索

% (パーセント) ワイルドカードを使用して、パターン内の文字数を示します。例えば、 lastname という名前のフィールドから「r」で終わる記録を全て抽出します。

```
lastname = "%r"
```

XML データベースデータの保管とクエリをサポートするデータベース (例えば、IBM DB2、Oracle、SQL Server) をクエリする場合、その特定のデータベースに適用することのできるXML 関数をキーワードを使用することができます。例 :

```
xmlexists('$c/Client/Address[zip>"55116"]' passing
USER.CLIENTS.CONTACTINFO AS "c")
```

次も参照してください:サンプル:IBM DB2 XML 型列からデータを抽出する

任意で特定のフィールドにより抽出されたレコードセットを並べ替えるには、「SQL WHERE/ORDER プロパティ」ダイアログボックスの対応するテキストボックスにORDER BY 句 を追加します。複数のフィールドを並べ替えるには、フィールド名 をコンマで区切ります。並べ替えの順序を変更するには、ASC とDESC キーワードを使用します。例えば、次のORDER BY 句は、 lastname とfirstname により降順に順序付けられたレコードを抽出します :

```
lastname, firstname DESC
```

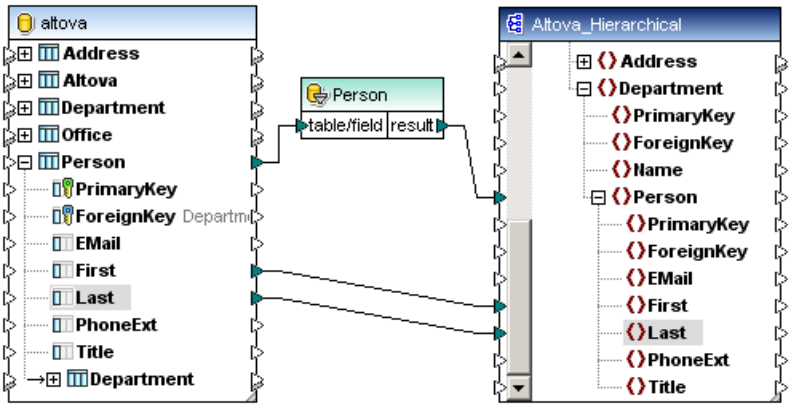
サンプル 1

次のWHERE 条件は **altova.mdb** データベースコンポーネントのPerson テーブルに添付されています。これは First とLast が文字 「C」より大きいレコードを抽出します。すなわち、「Callaby」以降の名前をすべて抽出します。SELECT 文によりFirst ならびにLast 内にあるコンテンツがC という文字より大きなレコードが取得されます。**altova.mdb** サンプルファイルのPerson テーブルにあるLast を例にすると、Callaby 以降のレコードが取得されます。

```
First > "C" AND Last > "C"
```

どのように接続が作成されているかに注目してください:

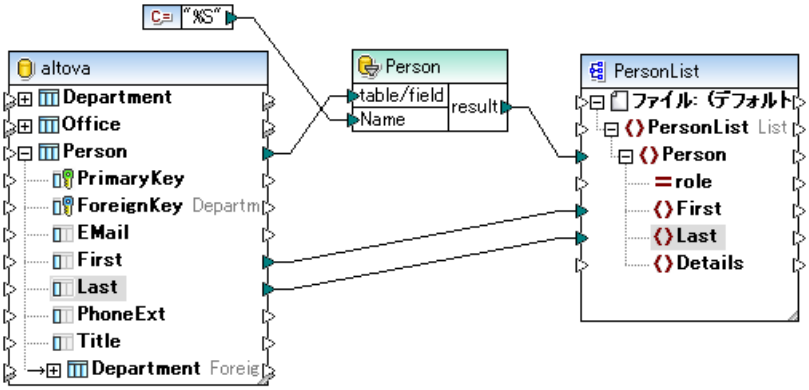
- 問い合わせを行いたいテーブル (この場合は「Person」) から**table/field** パラメーターへの接続を作成します。
- **result** パラメーターを、問い合わせ /フィルタリングを行ったフィールドの「親」アイテム (この場合はPerson アイテム) へ接続します。First ならびにLast のフィールドがターゲットコンポーネントに接続され、マッピングの結果に表示されます。



サンプル 2

このWHERE ステートメントにより、SQL WHERE/ORDER コンポーネントのパラメーターとして**Name** が作成されます。この場合、「Last」カラムにてパターンの検索が行われます。ワイルドカードの % により、任意の数の文字が指定されます。

```
Last LIKE :Name
```

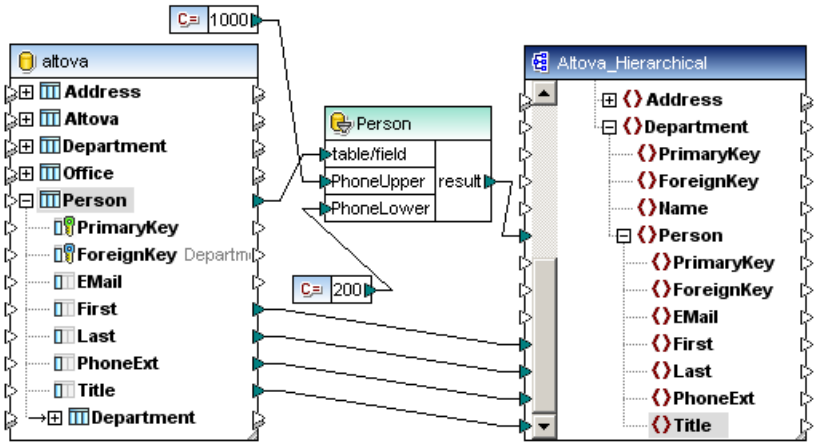


定数 コンポーネント%S は**Name** パラメーターの値を与えます。ワイルドカード % は文字数を表示します。これにより、マッピングが列 「Last」内のパターン(「S」で終わる名前)を検索します。

サンプル 3

このWHERE 句により、PhoneUpper とPhoneLower という 2 つのパラメーターが作成され、PhoneExt の値が比較されます。上限の値と下限の値は、以下のマッピングに示される通り、定数コンポーネントにより与えられます。

```
PhoneExt < :PhoneUpper and PhoneExt > :PhoneLower
```



この例にあるWHERE 句では、BETWEEN 演算子を使用することもできます

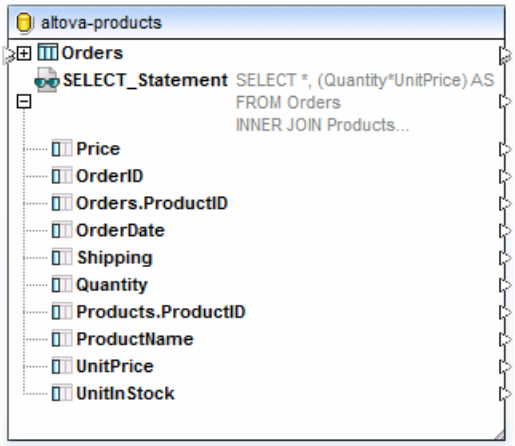
```
PhoneExt BETWEEN :PhoneUpper and :PhoneLower
```

7.2.7 仮想テーブルの SQL SELECT ステートメント

MapForce ではデータベースコンポーネントにてパラメーターを伴うSQL SELECT ステートメントの作成がサポートされています。SELECT ステートメントはテーブル型の構造をしており、SELECT ステートメントにより生成された結果のフィールドが含まれます。このような構造は、通常のデータベーステーブルやビューと同様に、マッピングデータソースとして使用することができます。

- SELECT ステートメントにて INNER/OUTER **JOIN** が使用されている場合、全てのテーブルのフィールドがコンポーネントには収められます。
- (SQL の "AS" キーワードにより指定された) 相関名を伴う表現もコンポーネント内のマッピングアイテムとして表示されます。
- SELECT ステートメントは既存のデータベーステーブルを MapForce へ挿入する際に作成されます。
- SELECT ステートメントには、SQL WHERE/ORDER コンポーネントで使用される構文によるパラメーターを含めることができます。

SELECT がデータベースコンポーネントに追加されると、返されるフィールドはマッピングのために使用することができます。例:



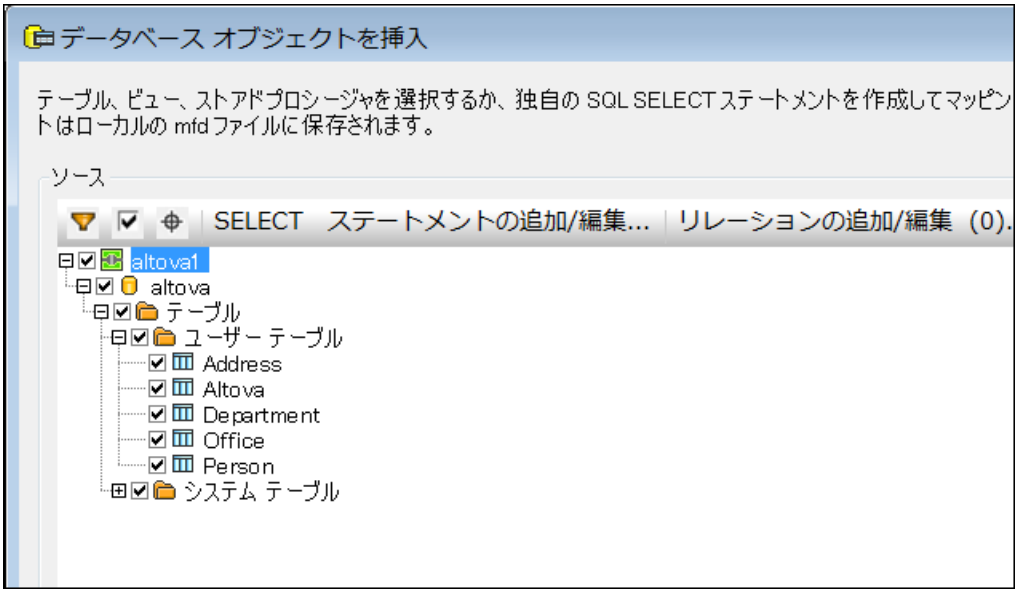
SELECT ステートメントはコンポーネント内部に表示されます。ステートメントにより表示される結果の行数は、メニューコマンド **「ツール | オプション」** を選択し、全般タブにて指定することができます。

7.2.7.1 SELECT ステートメントの作成

データベースコンポーネントを含むマッピングでSELECT ステートメントを作成することができます。マッピングがデータベースをまだ含んでいない場合、データベースを先に追加します（データベースへ接続するを参照）。このサンプルのスコープのために、メニューコマンド **挿入 | データベースを挿入** を選択して、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内で使用することのできる**altova-products.mdb** ファイルにウィザードに従い接続します。

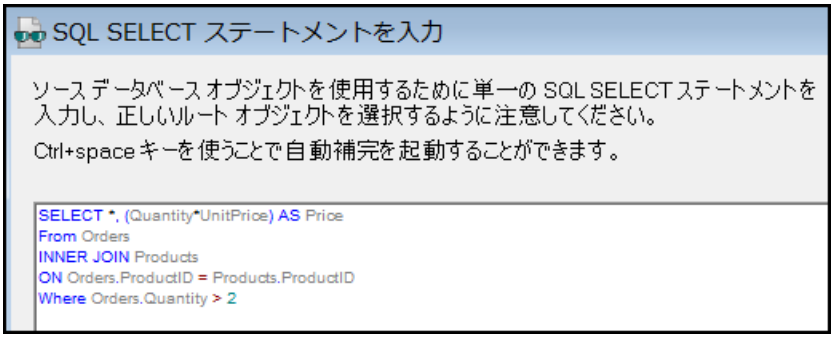
SELECT ステートメントの作成 ：

1. データベースコンポーネントのタイトルを右クリックして、 **追加 削除 データベースオブジェクトの編集** を選択します。（代わりに、データベースコンポーネントを選択してメニューコマンド **「コンポーネント | 追加 削除 データベースオブジェクトの編集」** を選択します。）



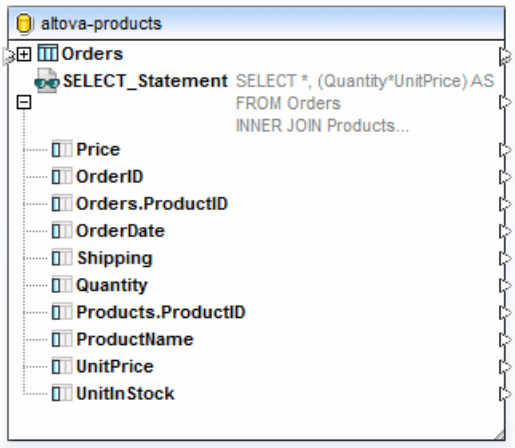
2. 以下を行います：
   - o 既存のテーブルからSELECT ステートメントを生成するには、テーブルを右クリックして、コンテキストメニューから、**SQL ステートメントの生成と追加** を選択します。生成されたステートメントは後に編集することができます。
   - o カスタムSELECT ステートメントを作成するには、 **SELECT ステートメントの追加 編集** ボタンをクリックします。
3. 必要に応じて、ステートメントを編集または作成します。例：下のSELECT ステートメントは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内の**altova-products.mdb** ファイルにとり有効です。 **Price** フィールドは、**Quantity** と**UnitPrice** の2つのフィールドで構成されており、関連付けられた名前（ `AS Price` として宣言されます。

```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```



4. **「SELECT ステートメントの追加」**をクリックします。SELECT ステートメントは、テーブル、ビュー、プロシージャと同様にデータベースオブジェクトとして、表示されます。

5. **「OK」**をクリックします。データベースコンポーネントにSELECT ステートメントが表示され、SELECT クエリにより返されたフィールドのデータをマップすることができます。



重要な注意点：

- SELECT ステートメント内のすべての計算された式は、マップすることのできるアイテムとして(このサンプルでは、"AS Price" のような)一意の関連付けられた名前を持つ必要があります。
- JDBC を使用して、Oracle または IBM DB2 データベースに接続する場合、SELECT ステートメントでは最後のセミコロンは存在しません。

SELECT ステートメントを削除する：

1. データベースコンポーネントのタイトルを右クリックして、**「追加 削除 データベースオブジェクトの編集」**を選択します。
2. 削除するSELECT ステートメントを右クリックして、**「SELECT ステートメントの削除」**を選択します。

7.2.7.2 SELECT ステートメントのパラメーター

このサンプルは、Microsoft Access データベースからデータを読み込み CSV ファイルに書き込む MapForce マッピングの作成方法について説明します。特に、この例のマッピングは、パラメーターを持つカスタムデータベースSELECT クエリ

を使用しています。 SELECT ステートメントは複数のテーブルからのデータを結合します。そして、更なる処理のために結果はマッピングへ与えられます。

このサンプルは、以下のパスで見つけることのできるマッピングデザイン (mfd) が伴います :**<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial\select-component.mfd**。このサンプルファイルを開き、最初に分析する、または、以下のステップを最初から行います。

このサンプルはMicrosoft Access データベースを使用しますが、その他のデータベース方でも同じ方法で作動します。データベースへ接続するを参照してください。

目的は、以下のとおりです :

1. データベースから、オーダーされるアイテムの数が、カスタムの値を超えるオーダーのみを選択する必要があります。このカスタムの値は、マッピングへのパラメーターとして与えられます。この目的を達成するために、入力パラメーターを持つカスタムデータベースSELECT ステートメントを作成します。
2. Access データベース内では、日時フォーマットは、`YYYY-MM-DD HH-MI-SS` です。CSV ファイル内では、時刻の部分が省かれるため、フォーマットは`YYYY-MM-DD` になります。この目的を達成するために、MapForce 内で使用することのできる、date-from-datetime 関数を使用します。
3. 結果 CSV ファイルは**OrdersReport.csv** という名前を持つ必要があります。

ステップ 1 :SELECT 構造を追加する

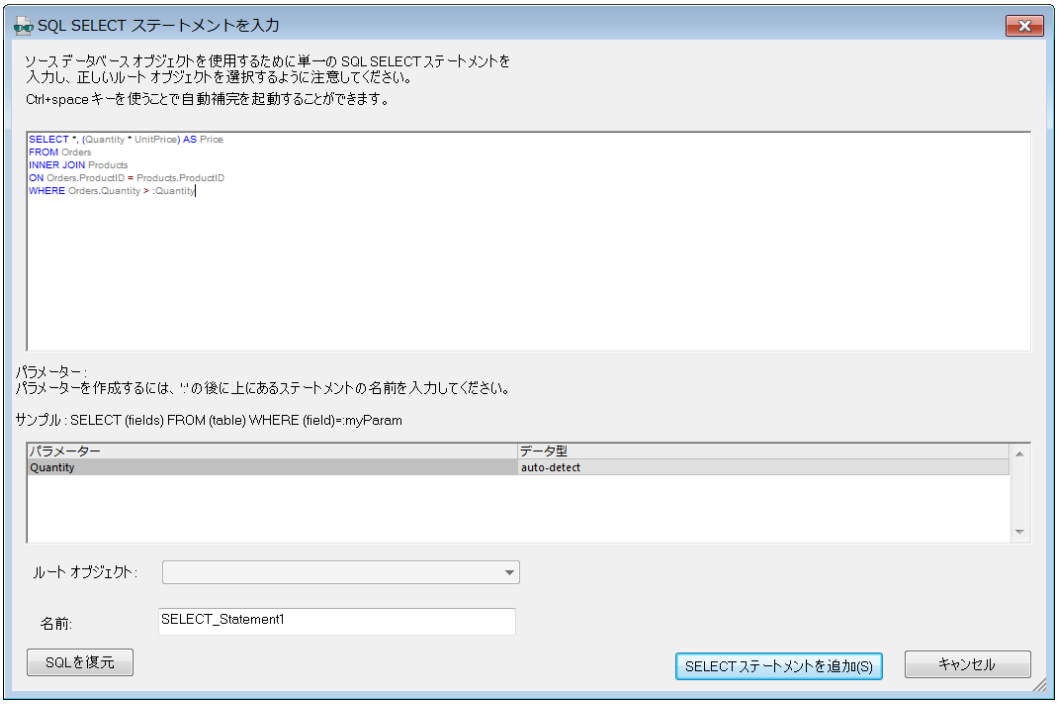
1. メニューから**データベースの挿入**をクリックします。
2. **Microsoft Access (ADO)** を選択して、**<Documents>\Altova\MapForce2017 \MapForceExamples\Tutorial** フォルダー内で使用することのできる**altova-products.mdb** ファイル**に接続するウィザードステップに従います。**



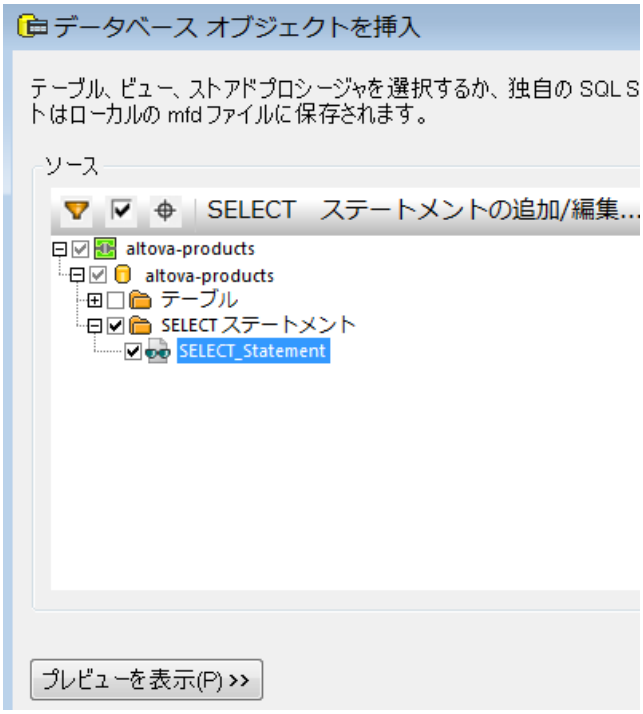
3. 「データベースオブジェクトの挿入」ダイアログボックスから **「SELECT ステートメントの追加 編集」**をクリックし、以下の クエリを挿入します :

```
SELECT *, (Quantity * UnitPrice) AS Price
FROM Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
WHERE Orders.Quantity > :Quantity
```

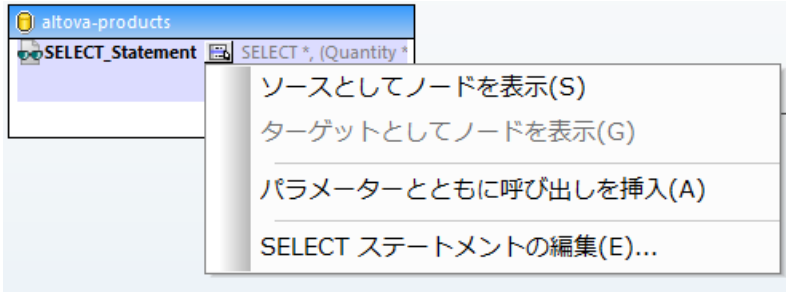
このクエリOrders とProducts テーブルの結合を使用し、すべての フィールド (*), と計算された値 (AS Price) 。クエリは以下も指定します : WHERE 句内の**Quantity** パラメーター



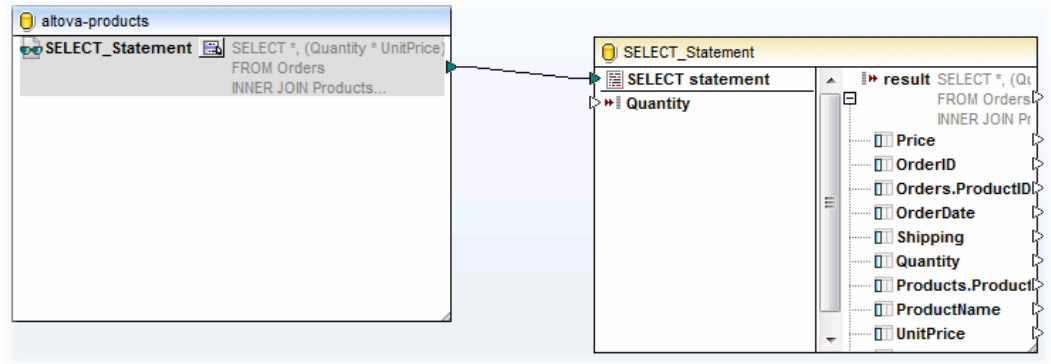
4. **「SELECT ステートメントの追加」**をクリックします。



5. 「**OK**」をクリックします。**altova-products** コンポーネントがマッピングエリアに追加されます。
6. **altova-products** コンポーネント から、**パラメータと共に呼び出しを挿入** を選択します。

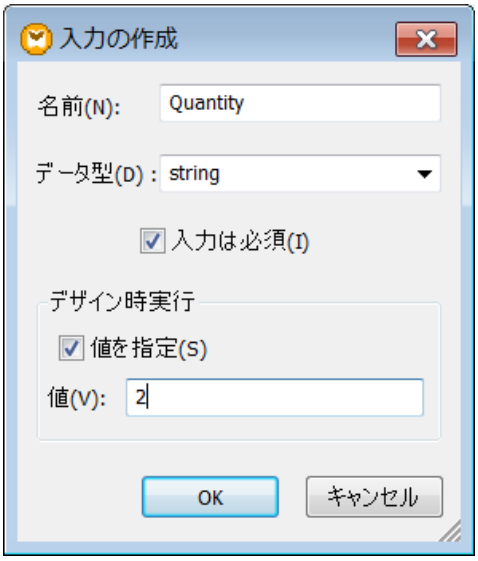


新規の構造（SELECT_ステートメント）をマッピングで使用することができるようになります。これは 2 つのパートに分割されます：左のパートは入力コネクタを、右のパートは出力コネクタを提供します。左のパートは前に定義された**Quantity** パラメーターを含んでいることに注意してください。

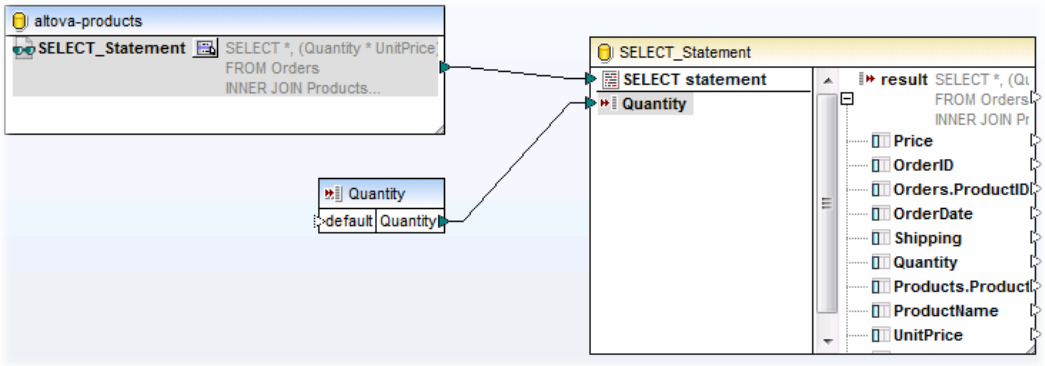


ステップ 2：入力パラメーターの追加

1. **挿入**」メニューから **入力の挿入**」をクリックします。
2. 型 "Quantity" を名前とします。
3. **Design-time Execution** 内に、デザイン中にマッピングの実行のために使用されるパラメーター値を入力します。 (このサンプルでは、 "2")詳細に関しては、マッピングにパラメーターを与えるを参照してください。



入力パラメーターをデータベースの呼び出しに接続します。構造は、以下のとおりです。

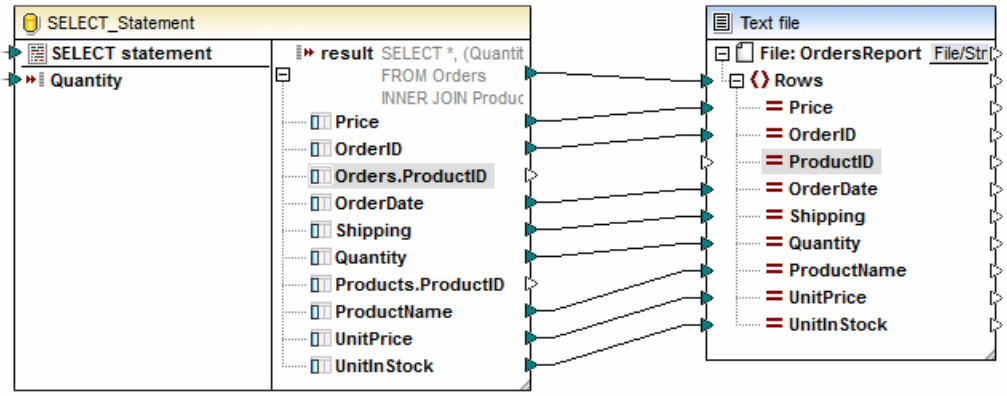


ステップ 3：ターゲット CSV コンポーネントを追加する

1. **挿入**」メニューから **テキストファイル**」をクリックします。
2. **標準 CSVのための単純な処理を使用する ..**」を選択し、 **続ける**」をクリックします。
3. コンポーネント設定ダイアログボックスから、 **フィールドの追加**」をクリックし、9つの新しいフィールドを追加します。データベースフィールドの名前と同じ名前をCSV フィールドに与えることが奨励されています。これは、マッピング接続を作成する際に時間の節約になります。これらの設定に関しての詳しい情報は、CSV オプションの設定を参照してください。



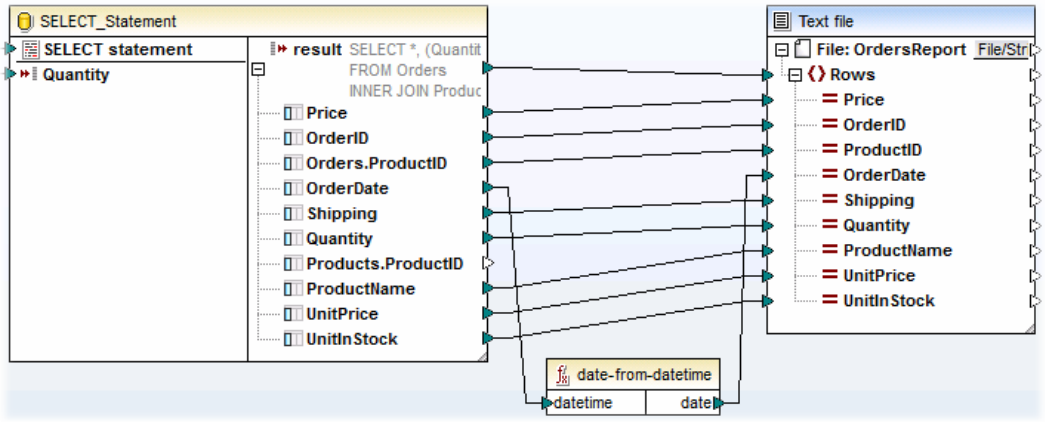
4. SELECT 構造の**result** ノードとCSV コンポーネントの**Rows** ノード間の接続を作成します。



CSV コンポーネントのフィールドの多くは、データベースと同じ名前を持ち、マッピング接続は、**result** を**Rows** に接続すると自動的に作成されます。これが発生すると、**「接続」**メニューを選択して、**「マッチする子を自動的に接続する」**オプションが有効化されていることを確認してください。SELECT 構造内でこの名前を持つフィールドがないため、手動で接続したマッピングアイテムは**ProductID** のみです。

ステップ 4：日付の変換

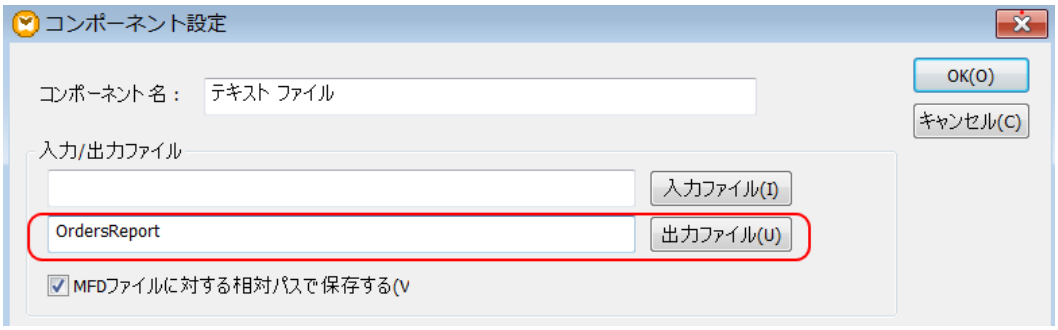
ライブラリウィンドウ内で、**date-from-datetime** 関数を検索し、マッピングエリアにドラッグします。入力と出力下で表示されるように接続します。



ステップ 5：出力ファイルの名前の設定

OrdersReport.csv の出力ファイルの名前を設定するには、CSV コンポーネントをダブルクリックし、出力ファイルボッ

ダイアログ内に値を入力します。



7.2.8 XML データから / データベースフィールドへのマッピング

MapForce を使用して、XML を保管するデータベースフィールド (列) からデータをマップ、またはデータベースフィールドへデータをマップすることができます。データベースフィールド (列) に保管される XML データが、抽出され、MapForce によりサポートされる他の構造に書き込まれること、またはその逆が可能になります。データを以下のようにマップすることができます：

1. XML 型専用の"へ"または"から"フィールド (例えば、SQL Server 内の `Xml`、Oracle 内の `XMLType`)。繰り返しまたは、またはXML へから書き込む専用のXML フィールドは、XML をサポートするネイティブのデータベースに適用することができます (IBM DB2、Oracle、および SQL Server など)。
2. XML コンテンツを保管する"へ"または"から"テキストフィールド (例えば、`Text`、`Varchar`)。これはテキストフィールドが XML ドキュメントを保管するために十分な長さがあるデータベースに適用することができます。

いずれにせよ、有効な XML スキーマは、マップするデータのデータベース列、またはデータベース列からのために存在する必要があります。データベース列が XML を保存する場合、MapForce は、(データベースがサポートする場合) XML スキーマを直接データベースから 割り当てる、または、外部ファイルからスキーマを選択するオプションを与えます。各データベース列につき、1つの XML スキーマを割り当てることができます。スキーマに複数のルート要素が存在する場合、スキーマの単一のルート要素を選択することができます。

XML がデータベース内で文字列フィールドとして保存されている場合、XML ドキュメントの文字エンコードは文字列フィールドをベースとしてます。データベースフィールドがテキストを Unicode として保存していない場合は、文字の一部を表示できない場合があります。

データベースの一部では、XML フィールドのための XML エンコード (このエンコードはデータベースの文字のセットと異なる文字のセットを有する場合があります) をサポートします。データベースによりサポートされている場合は XML ドキュメント エンコード宣言は XML フィールド内で宣言されているものと解釈されます。データベースにより与えられた XML エンコードサポートに関する詳細は、それぞれのドキュメントを参照してください。

7.2.8.1 XML スキーマをデータベースフィールドに割り当てる

このトピックはデータベース内で XML 型として定義されているフィールドにスキーマを割り当てる方法を説明します。下の説明は、SQL Server 2014 とAdventure Works 2014 データベースを使用しています。後者は、CodePlex Web サイト (https://msftdbprodsamples.codeplex.com/) からダウンロードすることができます。XML フィールドからの、またはXML フィールドへのデータのマッピングは、XML フィールドをサポートする他のデータベースの型と同様に動作します。

Adventure Works 2014 データベースをマッピングコンポーネントとして追加する：

1. **挿入** メニューから**データベース**をクリックし、接続ウィザードを優先順序 (ADO または ODBC) に従い進めます。更に詳しい情報に関しては、Microsoft SQL Server (ADO) に接続 と Microsoft SQL Server

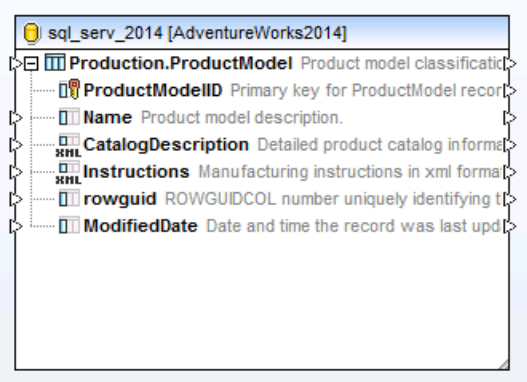
(ODBC) に接続 を参照してください。 メモ:**SQL Server Native Client** ドライバーを使用する場合、**統合されたセキュリティ**プロパティをスペース文字に設定する必要があります (SQL Server Data Link プロパティのセットアップを参照)。

2. **「データベースオブジェクトの挿入」** ダイアログボックスから、**Production** スキーマを展開し、**ProductModel** テーブルを選択します。

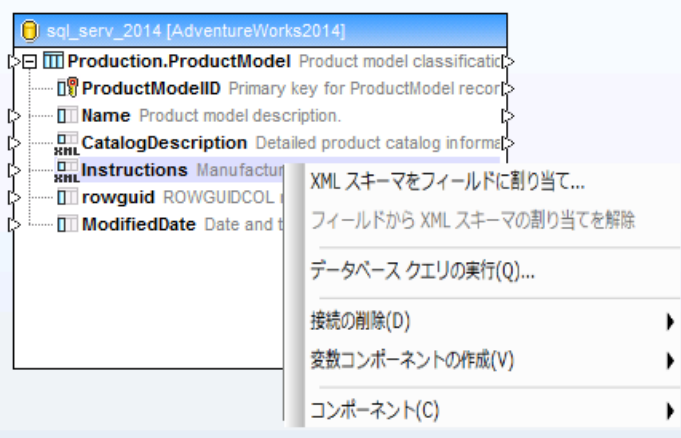


3. 「OK」をクリックします。

データベーステーブルがマッピングエリアに追加されました。このテーブルには次の2つの XML 型のフィールドがあります : **CatalogDescription** と**Instructions**:

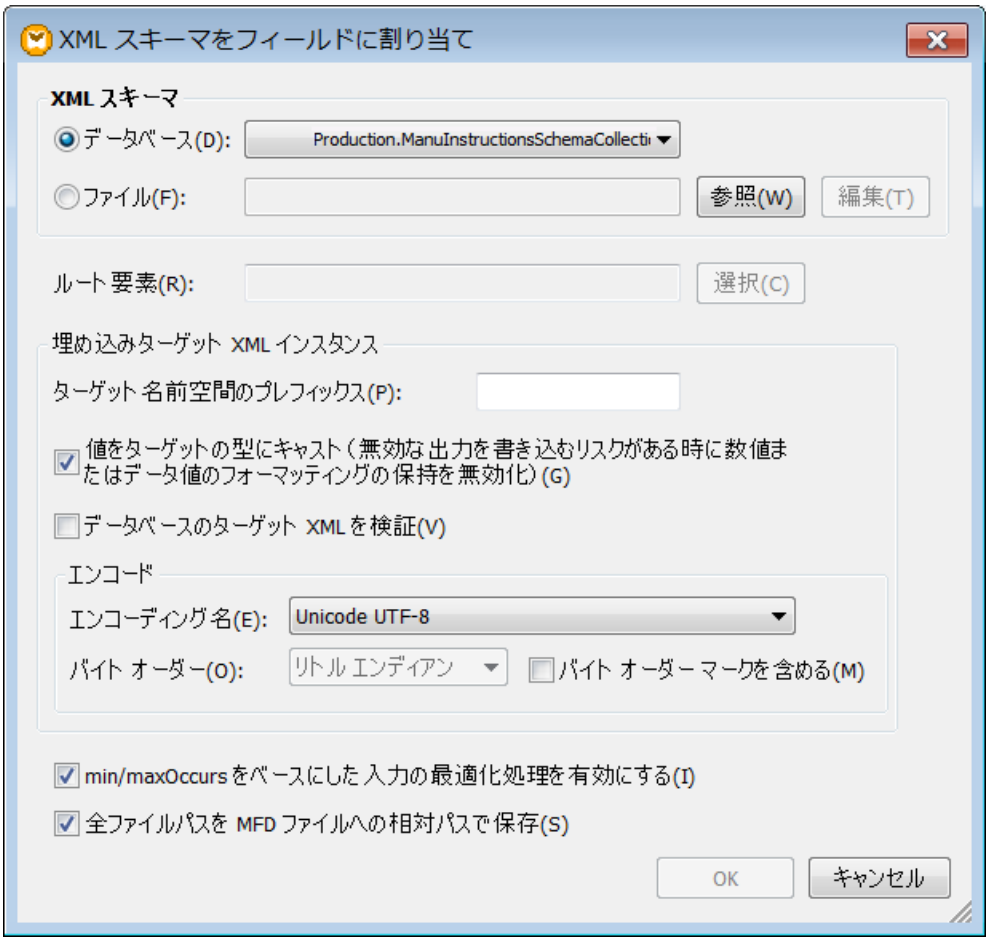


マッピング上に XML フィールドの構造が表示されるようにするには、フィールドコンテンツの XML スキーマが必要になります。**Instructions** フィールドを右クリックして、 コンテキストメニューから、**XML スキーマをフィールドに割り当てる**を選択します。

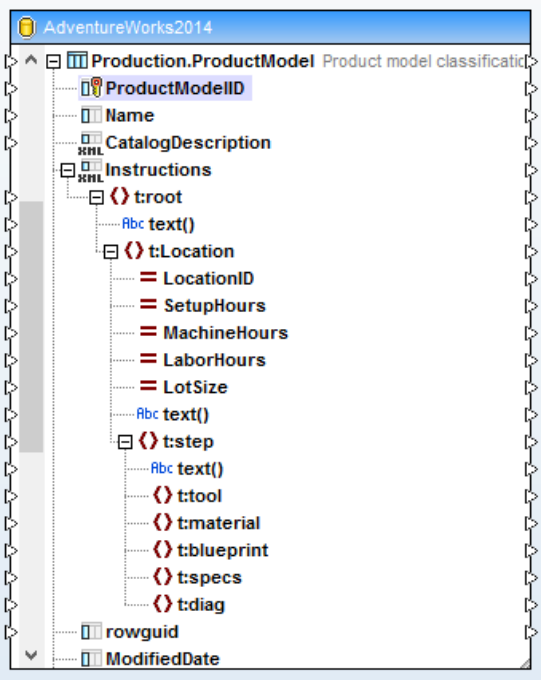


この特定の例の中では、**Instructions** フィールドに直接データベースからスキーマを割り当てます。これを行うには、**データベース**オプションの横の**Production.ManuInstructionsSchemaCollection** アイテムを選択し、 **「OK」**を

クリックします。



XML フィールドの構造がコンポーネント上に表示されます。 このフィールドへ、または、このフィールドへ接続を描く(およびマップを作成する)ことができます。



7.2.8.2 サンプル XML データを SQLite フィールドに書き込む

この例は、複数の XML ファイルからデータを読み込み、SQLite データベースに書き込む MapForce マッピングを作成する必要な手順について説明します。マッピングの目的は、それぞれのソース XML ファイルのために、SQLite データベース内に新規のデータベースを作成することです。それぞれのレコードは、TEXT フィールドとして XML ドキュメントを保管します。

使用されているサンプルは、以下のパスで見つけることができます：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ファイル名は、以下のとおりです：

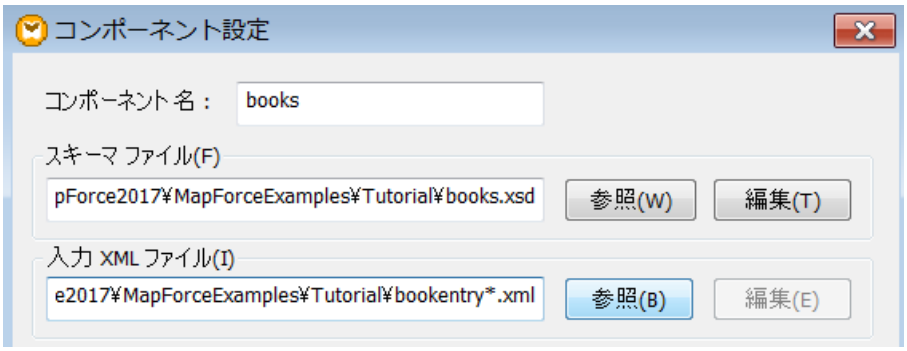
マッピングデザインファイル	• **XmlToSqliteField.mfd**
ソース XML ファイル	• **bookentry1.xml** • **bookentry2.xml** • **bookentry3.xml**
検証に使用される XML スキーマ	• **books.xsd**
ターゲット SQLite データベース	• **library.sqlite**

マッピングのゴールを達成するには、以下のステップを踏みます：

1. XML コンポーネントを追加し、複数のファイルから読み込むように構成します。
2. SQLite データベースコンポーネントを追加し、XML スキーマをターゲット TEXT フィールドに割り当てます。
3. マッピング接続の作成とデータベース「アクションの挿入」を構成します。

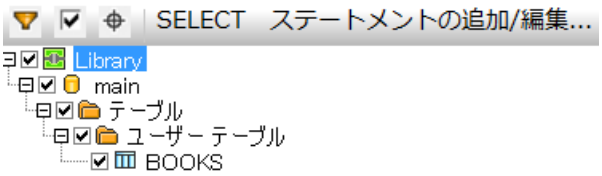
ステップ 1：XML コンポーネントの追加

1. **挿入** メニューから **XML スキーマ/ファイル** をクリックし、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリ内の **books.xsd** スキーマを参照します。XML ファイルを提供するようにプロンプトされると、**「スキップ」** をクリックします。ルート要素を選択するようにプロンプトされると、**Books** を選択します。
2. コンポーネントヘッダーをダブルクリックして、**入力 XML ファイル** ボックス内に **bookentry*.xml** を入力します。これにより、MapForce にソースディレクトリ内の、"bookentry-" から名前が始まるすべての XML ファイルから読み込むように命令します。この技術に関する詳細については、複数の入力または出力ファイルを動的に処理するを参照してください。

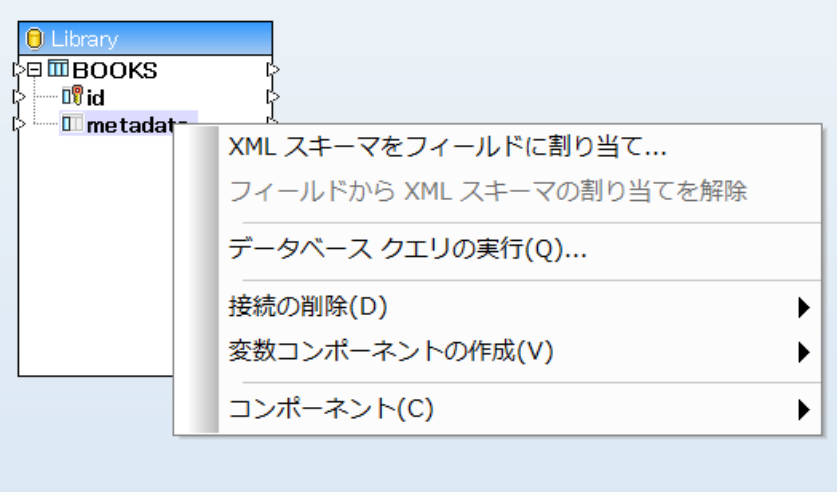


ステップ 2：SQLite コンポーネントを追加する

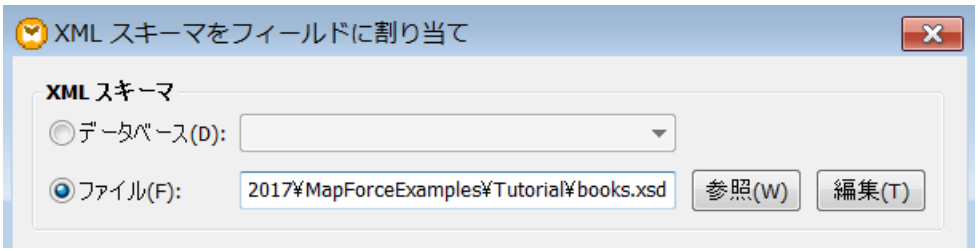
挿入 メニューから、**データベース** をクリックし、ウィザードに従い、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリの **Library.sqlite** データベースファイルに接続します（既存の SQLite データベースに接続するも参照してください。）。データベースオブジェクトを選択するように問われると、BOOKS テーブルを選択します。



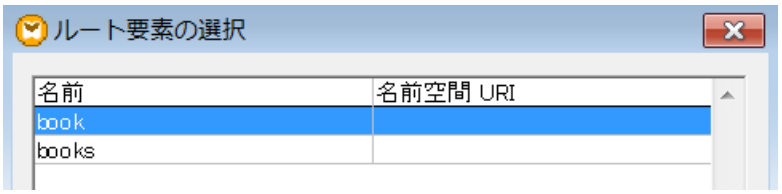
XML コンテンツが書き込まれるデータベースフィールドは、`metadata` と呼ばれます。このフィールドに XML スキーマを割り当てるには、右クリックして、コンテキストメニューから **XML スキーマをフィールドに割り当てる** を選択します。



このチュートリアルで、etadata フィールドに割り当てられているスキーマは、ソースXML ファイルを検証するために使用されているものと同じです。 **参照...** をクリックし、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリから**books.xsd** スキーマを選択します:

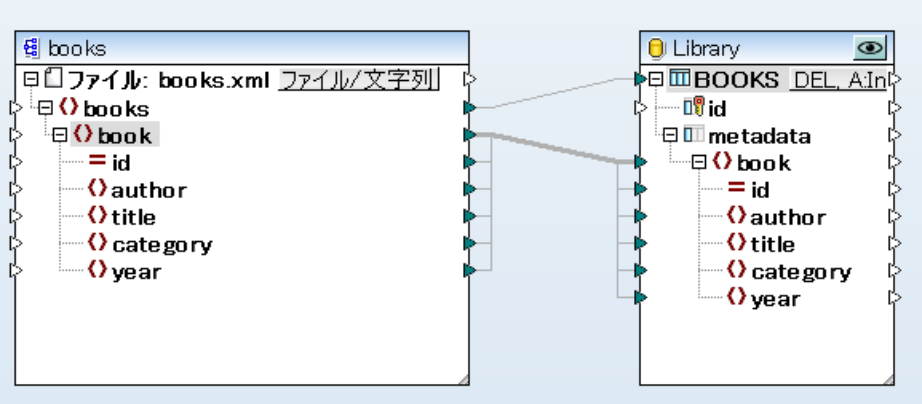


books.xsd スキーマには、次のグローバル宣言を持つ2つの要素があります :book とbooks。この例では、book をデータベースに書き込まれるXML のルート要素として設定します。**選択**をクリックして、book をルート要素として選択します。



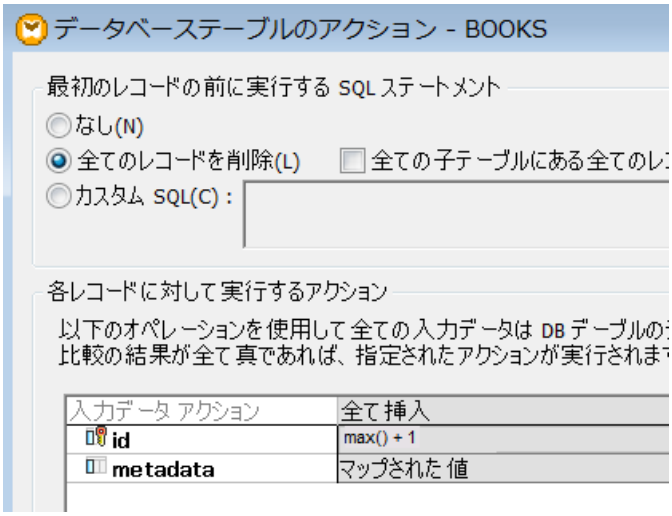
ステップ 3 : マッピング接続の作成とデータベース「アクションの挿入」の構成

マッピング接続の作成 は、以下の通りです :



上に示されるように、ソースとターゲットが同じスキーマと子要素の名前を使用するため、 book からbook への接続は、"すべてをコピー"接続です。このような接続に関する詳細については、すべてコピー接続を参照してください。

一番上の接続 (books からBOOKS) は、ソース内のそれぞれのbook 要素を反復し、新規のレコードをBOOKS テーブルに書き込みます。データベースコンポーネントの A:In ボタンをクリックして、データベースが、下で表示されるように設定を更新するように設定してください:



すべてのレコードを削除 オプションは、レコードを挿入する前にBOOKS テーブルのすべてのコンテンツを削除するようにMapForce に命令します。

すべて挿入 アクションは、データベースINSERT クエリが行われるように指定します。フィールドmetadata は、マッピングから提供される一方、フィールドid は、データベース自身から生成されます。

実行する前にマッピングが保存されていることを確認してください。

マッピングを実行し、生成された出力を確認するには、**出力** タブをクリックします。このアクションはデータベースをすぐに更新しないことに注意してください。生成されたデータベーススクリプトを実行する準備が整うと、メニューコマンド **出力| SQL スクリプトの実行** を選択します。(または ツールバーボタンをクリックします)。

7.2.8.3 サンプル: IBM DB2 XML 型列からデータを抽出する

このサンプルはXML 型のIBM DB2 データベース列からのデータの抽出方法、および、ターゲットCSV ファイルへの書き込みの方法について説明しています。また、条件に応じてXML コンテンツを抽出するするために、SQL 内に埋め込まれたXQuery ステートメントの使用方法についても説明されています。このサンプルは、テーブルを作成し、テーブルを表示する許可が存在する、IBM DB2 データベースへのアクセスが必要です。

XML データが実際に含まれるデータベースを準備します。使用中のデータベースに固有のデータベース管理ツール、または、MapForce で直接おこなうことができます。以下のステップを行ってください:

1. 新規マッピングを作成し、**DBQuery** タブをクリックします。
2. **「クイック接続」**( ) クリックして、ウィザードステップに従い、新規のデータベース接続を作成します (次を参照してください:データベース接続のサンプル).
3. SQL エディターに次のテキストを貼り付けます。このSQL クエリは、`ARTICLES` という名前のデータベーステーブルを作成し、データと共に表示します。

```
-- Create the table
CREATE TABLE
    ARTICLES (
        id INTEGER NOT NULL,
        article XML ) ;
-- Populate the table
INSERT INTO ARTICLES VALUES
   (1, '<Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>'),
(2, '<Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>230</SinglePrice>
   </Article>'),
(3, '<Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>'),
 (4, '<Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>5750</SinglePrice>
   </Article>');
```

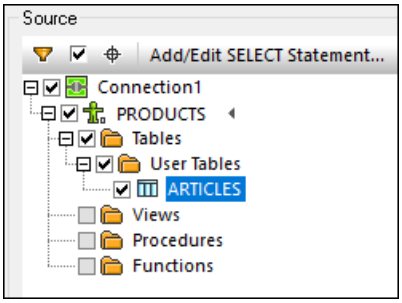
4. **「実行」**( ) ボタンをクリックします。クエリの実行の結果がクエリの結果ウィンドウに表示されます。クエリの実行に成功すると、4つの行が新規に作成されたテーブルに追加されます。

次に、条件に応じて作成された`ARTICLES` テーブルからXML データを抽出するマッピングを作成します。目的は、`ARTICLES` 列から100 よりも高額な商品のみを抽出することです。

ステップ 1:データベースを追加する

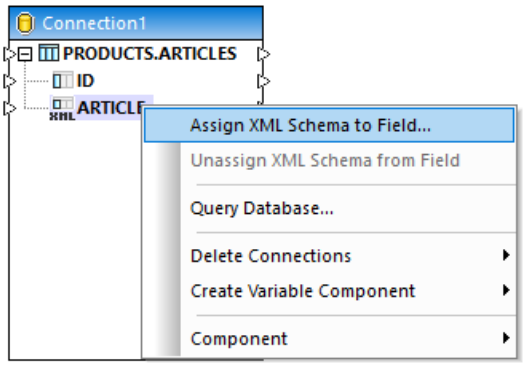
1. マッピングペインに切り替えるために **「マッピング」** タブをクリックします。
2. **「挿入」** メニューから **「データベース」** をクリックし、ウィザードステップに従い、データベースに接続します。

3. データベースオブジェクトを選択するようにプロンプトされると、以前に作成されている ARTICLES テーブルを選択します。



ステップ 2：フィールドに XML 型スキーマを割り当てる

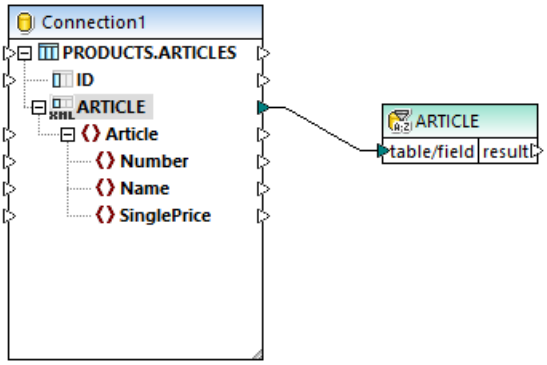
1. コンポーネントの ARTICLE アイテムを右クリックし、**フィールドに XML 型スキーマを割り当てる** をコンテキストメニューから選択します。



2. **「ファイル」** を選択して、次のスキーマを参照します：**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\DB2xsd.xsd**.

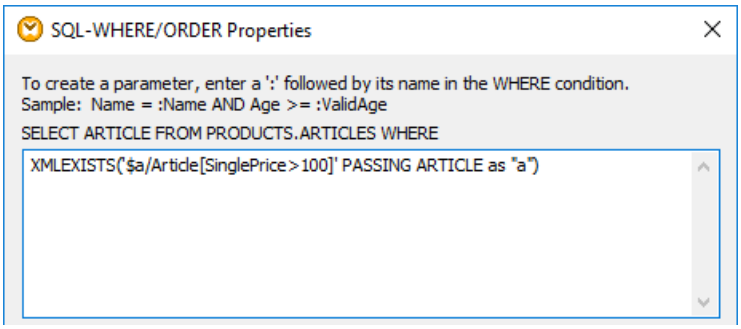
ステップ 3：SQL WHERE/ORDER コンポーネントを追加する

1. **「挿入」** メニューから **SQL WHERE/ORDER** をクリックします。
2. ARTICLE XML 型の列 SQL WHERE/ORDER の入力に接続します。



3. SQL WHERE/ORDER プロパティダイアログボックス内に次のテキストを入力します：

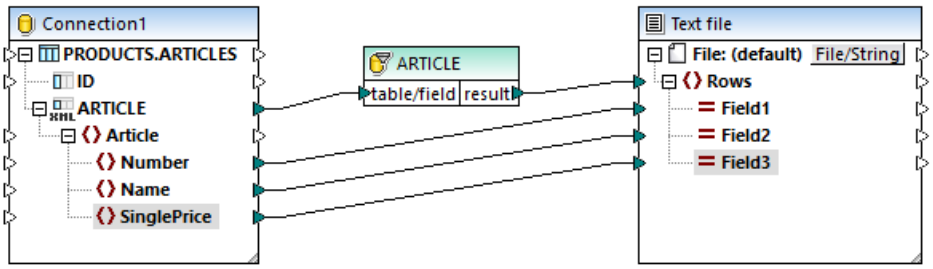
```
XMLEXISTS('$a/Article[SinglePrice>100]' PASSING ARTICLE as "a")
```



上のテキストは、SQL クエリの "WHERE" の部分を示しています。マッピングのランタイムでは、ダイアログボックス上で表示されている "SELECT" の部分と結合されています。このステートメントは、XMLEXISTS 関数とIBM DB2 データベースに固有の構文を使用しています。

ステップ 4：ターゲット CSV ファイルを追加する

1. **「挿入」**メニューから **「テキストファイル」**をクリックします。 .
2. プロンプトされると、 **標準 CSV のために簡単な処理を使用する..」**、 **「継続する」**を選択します。
3. **「フィールドの追加」**3回クリックして、商品番号、名前、価格をそれぞれ保管する3つのフィールドを追加します。他の設定を保留します。
4. 下に示されるようにマッピング接続を描きます。



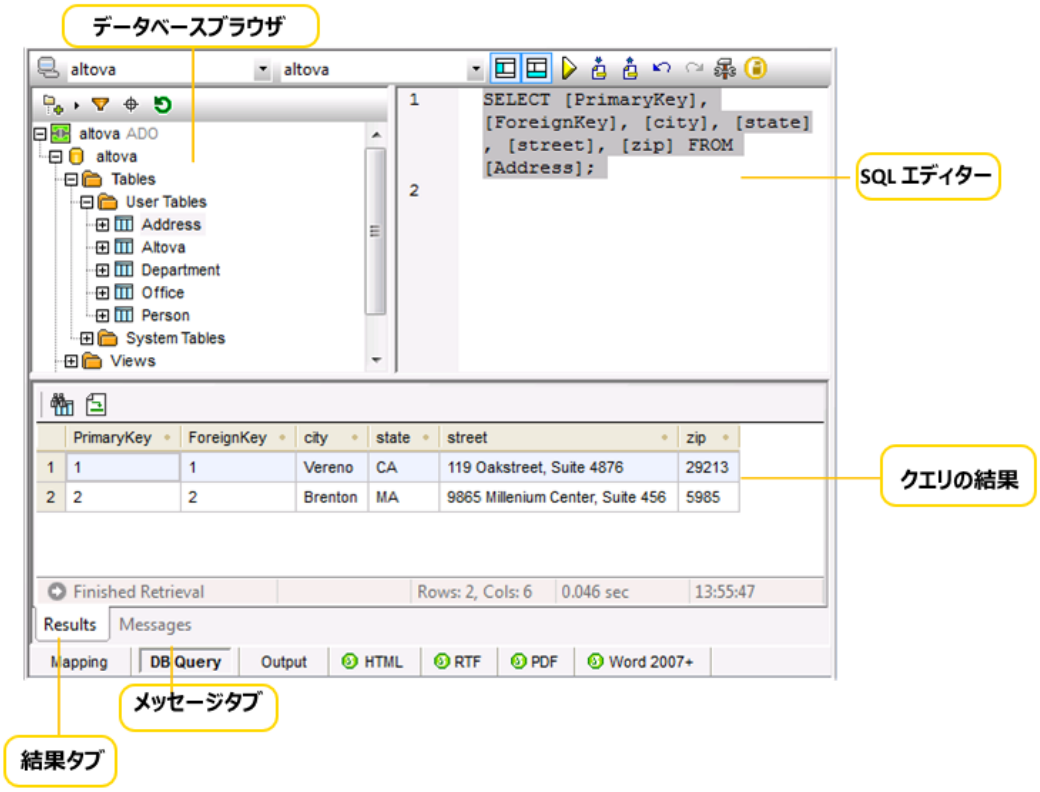
マッピングの結果を、**出力** タブをクリックして、プレビューすることができます。予想通り、出力内に 100 よりも高額の価格の商品のみが出力に表示されます。

```
1       2,Socks,230
2       4,Jacket,5750
```

7.2.9 データベースの参照とクエリ

MapForce にはマッピング処理から独立しデータベースをクエリすることのできる専用のデータベースクエリまたは**DB クエリ**ペインが搭載されています。このような直接的クエリは、マッピング *.mfd ファイルには保存されず、 MapForce から直接参照またはデータベースのコンテンツを変更することのできる便利な方法が与えられます。

個別の**DB クエリ**ペインは、現在アクティブなマッピングのそれぞれに存在します。**DB クエリ**ペイン内で異なるデータベースへの複数のアクティブな接続を作成することができます。**DB クエリ**ペインから作成された接続は、マッピングの一部ではなく、グローバルリソースとして定義しない限り MapForce を閉じた後には保管されません（グローバルリソース -データベースを参照）。



このタブのウィンドウは以下のパーツから構成されます：

- **データベースブラウザー** にはデータベーステーブルに対する接続情報が表示されます。
- **SQL エディター** ではSQL クエリを作成することができます。
- **結果タブ** にはクエリの結果がテーブル形式で表示されます。
- **メッセージタブ** には警告ならびにエラーメッセージが表示されます。

データベースクエリウィンドウの上部には接続コントロールが用意されており、作業用データベースや接続ならびにデータベーススキーマを定義することができます。

7.2.9.1 データベースの選択と接続

クエリするそれぞれのデータベースに対して、接続を作成する必要があります。マッピングが既にデータベースコンポーネントを含む場合、**DB クエリ** ペインの上のエリアにある既存のデータベース接続 (デフォルトでは接続はオフライン) を選択することができ、データベースオブジェクトを探索し、クエリを実行することができます。

マッピングがデータベースコンポーネントを含まない場合、または、新規のデータベースに接続する場合は、**クイック接続** ( ) をクリックし、新規のデータベース接続を作成するウィザードに従います (サンプル を参照)。接続の内の1つがグローバルリソースとして定義されている場合、グローバルリソースから既存のデータベース接続を選択することができます (グローバルリソース -データベースを参照)。

データベースに接続すると、次のメソッドを使用してデータベースクエリを作成することができます：

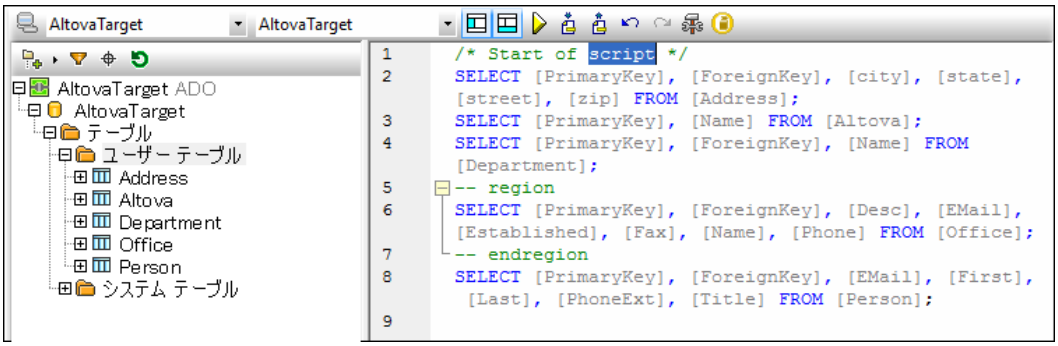
- SQL クエリをSQL エディター ペインから既存のSQL ファイルインポートする。
- クエリをSQL エディター ペイン内で作成する。
- データベースブラウザー ペイン内のオブジェクトを右クリックして、クエリを生成します。 (通常、SELECT)。

SQL エディターペイン内でクエリを実行する準備が整ったら、**実行** ボタンをクリックします。データベースデータはテーブルの様式で結果タブに表示されます。ステータスバーは検索の終了したメッセージ（検索の終了）を、およびクエリの結果に関連する適切な情報を表示します。

検索の終了メッセージが表示されると、検索結果を検索、並べ替え、クリップボードにコピーすることができます（データベースクエリ -結果 & メッセージタブを参照）。

7.2.9.2 SQL ステートメントの作成と編集

SQL エディターは、SQL ステートメントを作成または実行するために使用されます。既存の SQL スクリプトからコードされ自動的に生成された、または手動で作成されたすべての SQL ステートメントが表示されます。SQL エディターは自動補完、領域、またはラインブロックコメントをサポートします（次を参照してください：自動補完）。



SQL エディターツールバーで、以下のアイコンを使用することができます：

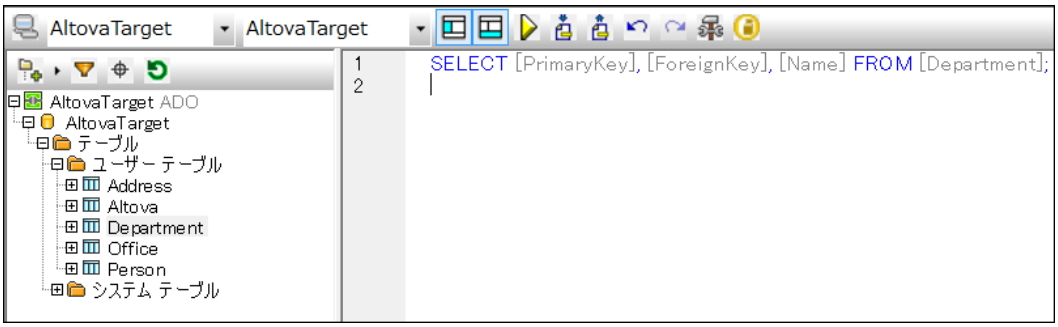
	ブラウザーの切り替え：ブラウザーペインの表示 / 非表示を選択します。
	結果の切り替え：結果ペインの表示 / 非表示を選択します。
	クエリの実行（F5）：このボタンをクリックすることで、現在選択されている SQL ステートメントが実行されます。複数のステートメントが表示されていて、そのうちどれも選択されていない場合、全てのステートメントが実行されます。
	元に戻す：SQL ウィンドウで行われた変更を、任意の回数だけ元に戻すことができます。
	やり直し：それまで元に戻した変更を再度入力することができます。元に戻すコマンドと併用することで、履歴内にある変更を確認することができます。
	SQL ファイルのインポート：SQL ファイルを SQL エディターにて開き、実行することができます。
	SQL ファイルのエクスポート：SQL クエリを保存して、将来も使用できるようにします。
	現在の SQL スクリプトを DatabaseSpy で開く：DatabaseSpy を起動し、SQL エディターウィンドウに表示されているスクリプトを開きます。
	オプション：オプションダイアログボックスを開き、MapForce 全般ならびに SQL エディターに関する設定を変更することができます。

SQL ステートメントの生成

SQL ステートメントスクリプトからロードされるはデータベースブラウザーから自動的に生成されることができ、また手動で入力することもできます。

データベースブラウザーからSQL SELECT ステートメントを生成するには、以下を行います：

- データベースブラウザー内の (テーブル または ビューなどの) データベースオブジェクトまたはフォルダーをクリックし、データベースオブジェクトをブラウザーペインからSQL エディターへドラッグします。



- データベースブラウザーからデータベースオブジェクトを右クリックして、**「SQL エディターで表示 | 選択」**を選択します。ブラウザーペインに表示されているデータベースオブジェクトを右クリックして、コンテキストメニューからオプションを選択

SQL ステートメントを手動で作成する：

1. SQL エディター内にSQL ステートメントを入力します。自動補完が自動的に生じる場合は、ステートメントの入力中にドロップダウンリストが表示されます。



2. .カーソルの**上**と**下**キーを使用して、提案ヒントを選択し、ハイライトされたオプションを挿入するために**Enter** を押します (以下も参照してください：SQL 自動補完の提案)。

SQL ステートメントの実行

SQL エディターに表示されるSQL ステートメントは、データベースに対して使用することができ、即時結果を確認することができます。SQL クエリと影響を受けた行は**DB クエリ**ペインの内 **メッセージ**ペインに表示されます。

複数のSQL ステートメントがSQL エディターに表示される場合、選択されたステートメントのみが実行されます。個々のSQL ステートメントを以下のように選択することができます：

- 左マウスボタンをクリックして、カーソルを特定のステートメントにドラッグします。
- SQL エディター内のライン番号をクリックします。
- 特定のステートメントをトリプルクリックします。

SQL ステートメントの実行：

1. SQL エディターにてSQL ステートメントを入力または選択します（SQL ステートメントの生成を参照）。
2. **クエリの実行** ボタンをクリックします。

SQL スクリプトのインポートとエクスポート

SQL エディターウィンドウに表示されているSQL を保存することで、スクリプトを将来再利用することができます。

SQL エディターペインに表示されている内容をファイルへ保存する：

- **SQL ファイルのエクスポート**アイコン をクリックして、保存するSQL スクリプトの名前を入力します。

過去に保存したSQL ファイルを開く：

- **SQL ファイルのインポート**アイコン をクリックして、SQL ウィンドウに表示するSQL ファイルを選択します。

SQL コメントの追加と削除

SQL エディターでは、ステートメントやステートメントの一部、またはステートメントのグループをコメントアウトすることができます。コメントアウトされたステートメントは SQL スクリプトの実行時にスキップされます。

テキストの領域でコメントアウトを行う：

1. ステートメントまたはステートメントの一部を選択します。
2. 選択されたステートメントを右クリックし、「**ブロックコメントを挿入/削除**」を選択します。

```
1
2     /*
3     SELECT [PrimaryKey], [ForeignKey], [city], [state],
      [street], [zip] FROM [Address];
4     SELECT [PrimaryKey], [Name] FROM [Altova];
5     SELECT [PrimaryKey], [ForeignKey], [Name] FROM
      [Department];*/
6
7     SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
      [Established], [Fax], [Name], [Phone] FROM [Office];
8     SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
      [Last], [PhoneExt], [Title] FROM [Person];
9
```

ラインでコメントアウトを行う：

- コメントアウトを行う位置を右クリックして、「**行コメントを挿入/削除**」を選択します。カーソルがある位置から、その行の最後までがコメントアウトされます。

ブロックコメントまたはラインコメントを削除する:

1. コメントアウトされた箇所を選択します。ラインコメントの削除を行う場合、コメントの前に挿入されているコメントマーク (--)だけを選択するだけでも同様の効果を得ます。
1. 右クリックして、「**ブロック (または行) コメントを挿入/削除**」を選択します。

ブックマークを使用する

*ブックマーク*を使用することで、*スクリプト内にあるアイテム*をマークすることができます。

ブックマークを追加する:

- *ブックマーク*を行いたい行を右クリックして、コンテキストメニューから*ブックマーク*の挿入 削除を選択します。

```
1
2     SELECT [PrimaryKey], [ForeignKey], [city], [state],
      [street], [zip] FROM [Address];
3     SELECT [PrimaryKey], [Name] FROM [Altova];
4     SELECT [PrimaryKey], [ForeignKey], [Name] FROM
      [Department];
5     SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
      [Established], [Fax], [Name], [Phone] FROM [Office];
6     SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
      [Last], [PhoneExt], [Title] FROM [Person];
7
```

*ブックマーク*アイコン● が*ブックマーク*余白に表示されます。

ブックマークを削除する:

- *ブックマーク*を削除したい行をクリックして、コンテキストメニューから*ブックマーク*の**挿入/削除**を選択します。

ブックマーク間のナビゲーションを行う:

- 右クリックにて表示されるコンテキストメニューから、**次のブックマークへ移動**を選択することで次のブックマークへカーソルを移動します。
- 右クリックにて表示されるコンテキストメニューから、**前のブックマークへ移動**を選択することで前のブックマークへカーソルを移動します。

全てのブックマークを削除する:

- 右クリックして**全てのブックマークを削除**を選択します。

領域の挿入

領域とはテキストの選択範囲のことで、ユニット単位でSQL スクリプトのマーキングや宣言を行うことができます。領域は展開ならびに縮退することができ、SQL スクリプトの一部を隠すことができます。ある領域を、別の領域内部にネストすることもできます。

領域を挿入すると、選択されたテキストの上部に --region コメントが挿入され、展開 縮退アイコンが表示されます。

メモ: -- region コメントの後に領域の説明をしたテキストを挿入することで、領域の名前を変更することができます。"region" という名前を変更しないでください (例: --region DB2query)。

領域を作成する:

1. 領域として宣言したいステートメントをSQL エディターにて選択します。
2. 右クリックして表示されるコンテキストメニューから**領域の挿入**を選択します。マークされた箇所が領域になり、展開ならびに縮退を行うことができます。

```
1
2     SELECT [PrimaryKey], [ForeignKey], [city], [state],
      [street], [zip] FROM [Address];
3   -- region
4     SELECT [PrimaryKey], [Name] FROM [Altova];
5     SELECT [PrimaryKey], [ForeignKey], [Name] FROM
      [Department];
6   -- endregion
7     SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
      [Established], [Fax], [Name], [Phone] FROM [Office];
8     SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
      [Last], [PhoneExt], [Title] FROM [Person];
9
```

3. + または -ボックスをクリックして、領域の展開または縮退を行うことができます。

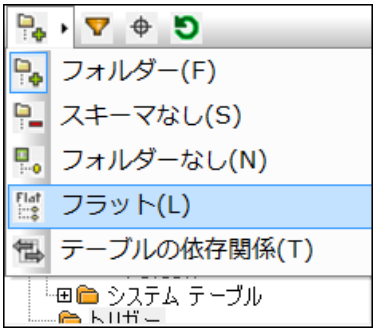
領域を削除する:

- -- region ならびに --endregion コメントを削除します。

7.2.9.3 データベースオブジェクトの参照

1つまたは複数のデータベースに接続している場合、**「データベースブラウザー」**ペインは、各データベース内のテーブル、ビュー、プロシージャ、などの詳細を含むオブジェクトの概要を表示します。XML をサポートするデータベースでは、**「データベースブラウザー」**で個別のフォルダー内の登録されたXML スキーマも追加して表示します。

データベースオブジェクトをナビゲートするには、**「データベースブラウザー」**ペインに複数の定義済みのデータベース表示レイアウトがあります。データベースブラウザーの上の部分で、定義済みのレイアウトを使用することができます。



レイアウトを選択するには、**フォルダーのレイアウト**( ) ドロップダウンボタンをクリックして、リストからエントリを選択します。このボタンは選択されたレイアウトを変更することに注意してください。

- **フォルダー** レイアウトは、データベースオブジェクトをオブジェクト型をベースにしたフォルダーを階層構造ツリーに整理します。これはデフォルトの設定です。

- **スキーマなし** レイアウトは、フォルダーレイアウトに類似していますが、データベーススキーマは存在しません。ですから、テーブルはデータベーススキーマ別に整理されません。
- **フォルダーなし** レイアウトはデータベースオブジェクトをフォルダーを使用せずに階層構造に表示します。
- **フラット** レイアウトは、データベースオブジェクトを最初の階層構造で型別に表示します。例えば、対応するテーブルに列が含まれるのではなく、すべての列が個別の列フォルダーに表示されます。
- **テーブルの依存関係** レイアウトは、テーブルを他のテーブルとの関係に基づき整理します。外部キーを持つテーブル、他のテーブルとの関係を持たないテーブルのためのカテゴリーが存在します。

レイアウトナビゲーションに加え、**「データベースブラウザー」**以下のタスクのために使用することができます：

- SQL ステートメントの生成（SQL ステートメントの生成 を参照）。
- 表示されるデータベースオブジェクトのフィルターおよび検索（データベースオブジェクトのフィルターおよび検索を参照）。
- "システム"と "ユーザー"テーブルにテーブルを並び替える
- アクティブなデータソースのルートオブジェクトの更新

ユーザーとシステムテーブルにテーブルを並び替える：

- **「データベースブラウザー」**内の "テーブル " フォルダーを右クリックして、**ユーザーとシステムテーブルの並べ替え**を選択します。

  メモ： この関数は、次のレイアウトが選択されていると使用することができます：**フォルダー**、**スキーマなし**、または **フラット**。

アクティブなデータソースのルートオブジェクトの更新：

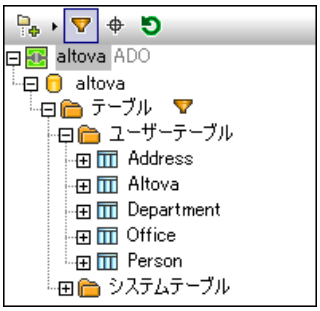
- **「データベースブラウザー」**の上にある、**更新**（ ）をクリックします。

データベースオブジェクトの フィルタリングと検索

「データベースブラウザー」に表示されるデータベースオブジェクト（スキーマ、テーブル、ビューなど）を名前または名前の一部によりフィルターすることができます。オブジェクトは文字を入力中にフィルターされます。フィルターはデフォルトで大文字と小文字を区別します。フィルターは "No Folders" レイアウトを選択している場合サポートされません。

データベースオブジェクトのフィルタリング

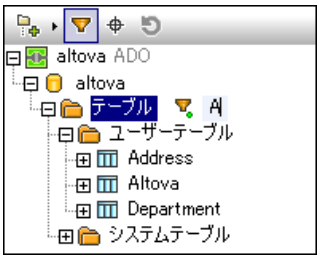
1. ツールバーから**フォルダーコンテンツのフィルターアイコン**（ ）をクリックします。現在のレイアウトで表示されているフォルダーの隣にフィルターアイコンが表示されます。



2. フィルタリングを行うフォルダーの隣にあるフィルターアイコンをクリックして、ポップアップメニューからフィルタリングオプション（例えば 含む）を選択します。



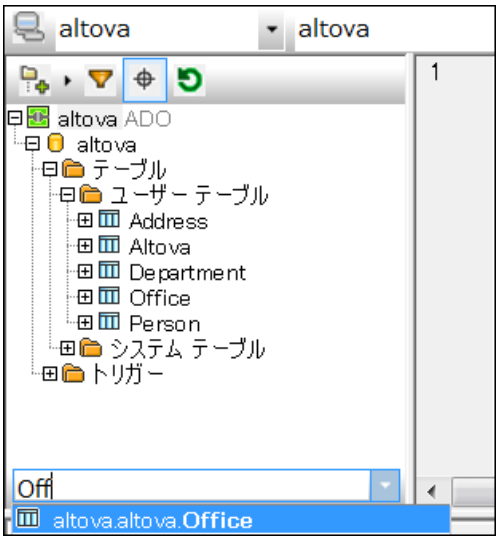
3. 空のフィールドがフィルターアイコンの隣に表示されます。検索を行う文字列（例：'G'）を入力します。入力中に検索結果が表示されます。



データベースオブジェクトの検索

名前によりデータベースアイテムの検索を行うには、フィルター機能の他に**オブジェクトロケーター**を使用することができます。

1. ブラウザーペインにて**オブジェクトロケーター**アイコン（ ）をクリックします。
2. ドロップダウンリストがブラウザーペイン下部に表示されます。検索を行う文字列を入力します（例：'Off'）。



3. **データベースブラウザー**内のリストからオブジェクトをクリックします。

データベースブラウザー内のコンテキストオプション

データベースブラウザー内で使用することのできるコンテキストメニューオプションは選択されたオブジェクトにより異なります。例えば：

- 「ルート」オブジェクトを右クリックすると、データベースを **更新** することができます。
- フォルダーを右クリックすると常に同じオプションが表示されます：**展開 | 兄弟 | 子** と **折りたたみ | 兄弟 | 子**。
- データベースオブジェクトを右クリックすると、下で説明される **SQL エディターで表示** コマンドとサブメニューアイテムが表示されます。

複数のデータベースオブジェクトを選択する場合、 **Shift + クリック** または **Ctrl + クリック** を押します。

メモ SQL ステートメントの構文は、使用するデータベースにより異なります。下の文法は Microsoft SQL Server 2014 に適用することができます。

ルートオブジェクトのために **SQL エディターで表示** コンテキストメニューの次のオプションを使用することができます：

- **CREATE**: 選択されたデータベースのルートオブジェクトのために CREATE ステートメントを作成します。例えば：
  ```
  CREATE DATABASE [MYDB]
  ```
- **DROP**: 選択されたデータベースのルートオブジェクトのために DROP ステートメントを作成します。例えば：
  ```
  DROP DATABASE [MYDB]
  ```

テーブルとビューのために **SQL エディターで表示** コンテキストメニューの次のオプションを使用することができます：

- **SELECT**: ソーステーブルの全ての列からデータを抽出し SELECT ステートメントを作成します。例えば：
  ```
  SELECT [DepartmentID],[Name],[GroupName],[ModifiedDate] FROM [MYDB].
  [HumanResources].[Department]
  ```
- **Name**: 選択されたテーブルの名前を返します。
- **Path**: テーブルのフルパスを以下のフォーマットで返します：¥
  ```
  DataSourceName.DatabaseName.SchemaName.TableName.
  ```

複数のテーブルを選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

列のために **SQL エディターで表示** コンテキストメニューの次のオプションを使用することができます：

- **SELECT**: 親テーブルの選択された列からデータを取得する SELECT ステートメントを作成します。 例えば：
  ```
  SELECT [DepartmentID] FROM [MYDB].[HumanResources].[Department]
  ```
- **Name**: 選択された列の名前を返します。
- **Path**: 列のフルパスを以下のフォーマットで返します：
  ```
  DataSourceName.DatabaseName.SchemaName.TableName.ColumnName.
  ```

複数の列を選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

制約のために **SQL エディターで表示** コンテキストメニューの次のオプションを使用することができます::

- **Name**: 選択された制約の名前を返します。
- **Path**: 制約のフルパスを以下のフォーマットで返します：
  ```
  DataSourceName.DatabaseName.SchemaName.TableName.ConstraintName.
  ```

複数の制約を選択した場合、 名前またはパスはコンマで区切られ、異なるラインに表示されます。

次のオプションを使用することができます。 **SQL エディターで表示** コンテキストメニューからインデックス：

- **Name**: 選択されたインデックスの名前を返します。
- **Path**: インデックスのフルパスを以下のフォーマットで返します：
  ```
  DataSourceName.DatabaseName.SchemaName.TableName.IndexName.
  ```

複数のインデックスを選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

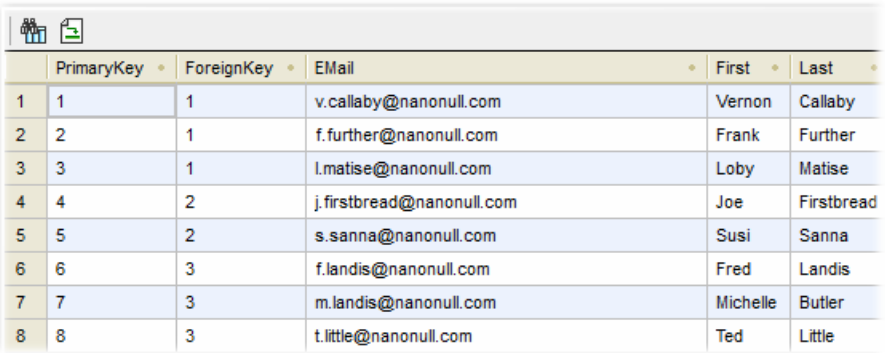
データベースが XML スキーマをサポートする場合、 "XML スキーマ" フォルダーに表示される全てのスキーマのために次のオ

プションを使用することができます：

- **XMLSpy 内のビュー**: XMLSpy 内でデータベーススキーマを開きます。
- **XML スキーマの管理**: 新規データベースXML スキーマを追加し、既存のデータベースXML スキーマを削除することのできるダイアログボックスを開きます。

7.2.9.4 クエリの結果のコピー、並べ替え、および検索

DB クエリペインの**結果**タブは、データベースクエリの結果として抽出されたレコードセットを表示します。



	PrimaryKey	ForeignKey	EMail	First	Last
1	1	1	v.callaby@nanonull.com	Vernon	Callaby
2	2	1	f.further@nanonull.com	Frank	Further
3	3	1	l.matise@nanonull.com	Loby	Matise
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread
5	5	2	s.sanna@nanonull.com	Susi	Sanna
6	6	3	f.landis@nanonull.com	Fred	Landis
7	7	3	m.landis@nanonull.com	Michelle	Butler
8	8	3	t.little@nanonull.com	Ted	Little

ツールバーにより、結果とSQL ステートメントの間をナビゲートすることができ、クエリの結果内の検索を促進します。

	**検索**：結果ウィンドウにある文字列を検索します。F3 キーを押下することで検索を継続することができます。
	**ステートメントへ移動**: SQL エディターウィンドウへ移動し、その結果を得るために使用されたSQL ステートメントのグループがハイライトされます。

クエリの結果からセルを選択する：

- 列ヘッダーをクリックして、全体の列を選択します。
- 列全体を選択するために、列番号をクリックします。
- 個別のセルをクリックします。複数の選択を行うことができる「**Ctrl**」キーを押します。列 またはセルがXML データを含む場合、このデータもコピーすることができます。

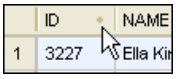
  メモ：コンテキストメニュー「**選択 | 行 | 列 | すべて**」もデータの選択に使用する子ができます。

クリップボードに選択されたセルをコピーする：

- 右クリックして、コンテキストメニューから「**選択されたセルをコピーする**」を選択します。

データの並べ替え：

- 並べ替えるために列を右クリックし、「**並べ替え | 昇順** または **降順**」を選択します。
- 列のヘッダー内の並べ替えアイコンをクリックします。



	ID	NAME
1	3227	Ella Kir

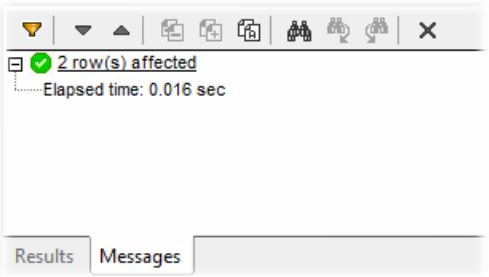
データは並べ替えられた列のコンテンツにより並べ替えられます。

デフォルトの並べ替え順序の復元：

- テーブル内を右クリックし、コンテキストメニューから **「並べ替え | デフォルトの復元」** を選択します。

7.2.9.5 実行されたクエリの状態を確認する

DB クエリペインの**メッセージ**タブは、最後に実行されたSQL ステートメントに関する特定の情報を提供し、また、エラーまたは警告メッセージを報告します。



メッセージタブのカスタマイズされた異なるフィルターを使用するか、または、**「次へ」**および**「前へ」**ボタンを使用して、データを行から行へと参照することができます。上のボタンは、メッセージをナビゲート、クリップボードへコピー、メッセージの一部を非表示するために使用されます。これらのオプションは、**メッセージ**タブ内を右クリックすると表示されるコンテキストメニュー内でも使用することができます。

	フィルター: 個別のメッセージのパート（**概要、成功、警告、エラー**）を開いた箇所からポップアップメニューを表示のために開きます。「自動挿入」は、SQL ステートメント、または、SQL 構造がSQL エディターにより自動的に挿入されると挿入されるメッセージを指します。「進行状況」メッセージは、データベース接続の結果、およびSQL 解析とデータ構造のロードの結果もを報告します。 **「全てチェックする」**または**「チェックを全て解除する」**をポップアップメニューから選択し、マウスをクリックするだけでこれらのオプションを全て選択する、または、選択しないことができます。
	次へ: 次のメッセージに移動してハイライトします。
	前へ: 前のメッセージに移動してハイライトします。
	選択されたメッセージをクリップボードにコピーします。
	選択されたメッセージとその子をクリップボードにコピーします。
	すべてのメッセージをクリップボードにコピーします。
	検索: **検索**ダイアログボックスを開きます。
	前を検索: **検索**ダイアログボックス内で指定された文字列の前の発生まで移動します。
	次を検索: **検索**ダイアログボックス内で指定された文字列の次の発生まで移動します。
	クリア: SQL エディターウィンドウのメッセージタブからすべてのメッセージを削除します。

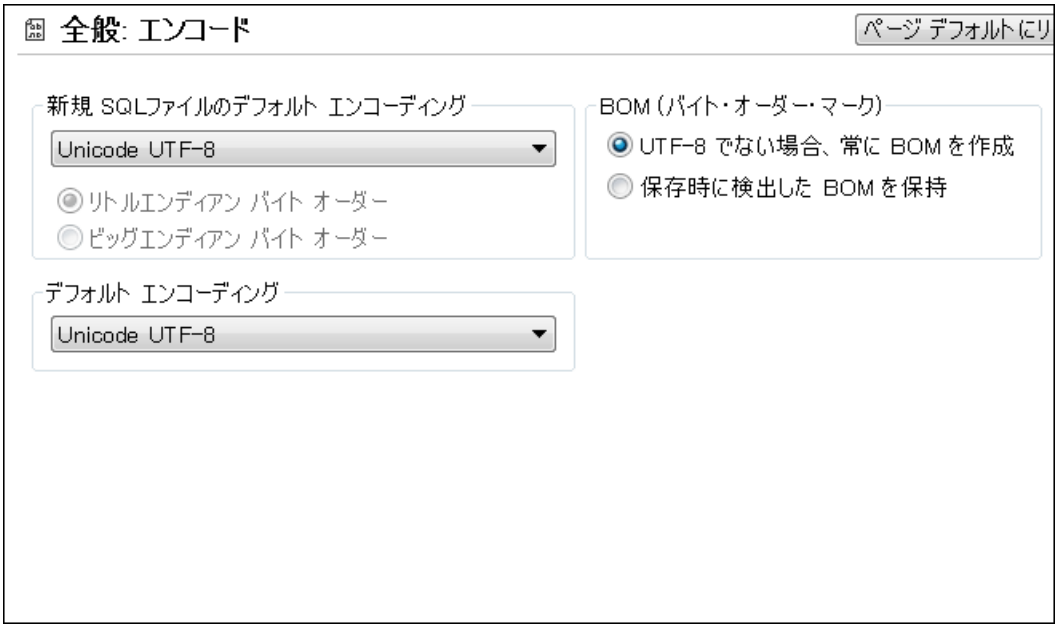
7.2.9.6 データベースクエリ設定

このセクションでは、入力された SQL ステートメント、または、ロードされた SQL エディターに適用することのできるその他の設定の構成についての情報と、クエリが実行された後に表示されるクエリの結果について説明されています。

SQL ファイルエンコードの設定

作成された SQL ファイルまたは SQL エディターで開かれたファイルのためのエンコードオプションの指定方法は、以下の通りです：

1. **「DB クエリ」** タブをクリックします。
2. ペインの上にある、**「オプション」** をクリックして、**「全般 | エンコード」** をクリックします。



新規 SQL ファイルのデフォルトエンコーディング

新規ファイルで使用するデフォルトのエンコーディングを指定することで、新たに作成されるドキュメントには、そのエンコーディング指定が追加されます。2 または 4 バイトのエンコーディング (UTF-16、UCS-2、または UCS-4) が選択された場合、リトルエンディアンまたはビッグエンディアンを選択することができます。

既存のファイルで使用されているエンコーディングは、勿論そのまま保持されます。

未知のエンコードを使用して SQL ファイルを開く

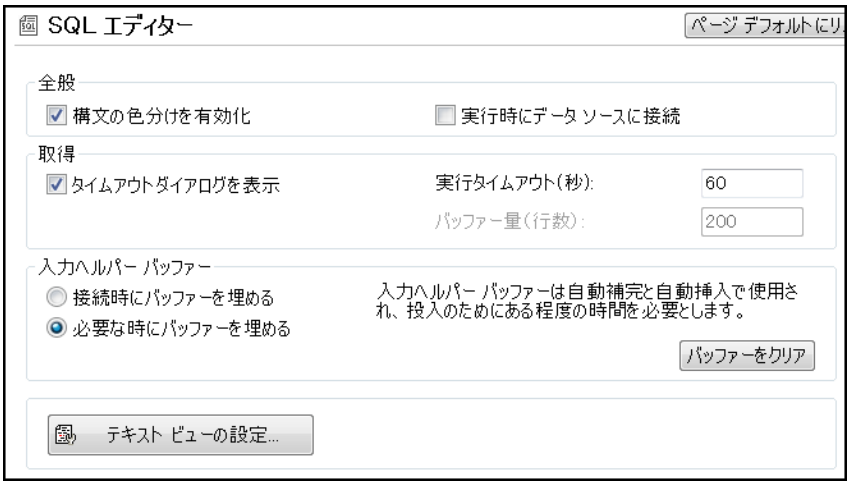
エンコードの指定がない、またはエンコードを検出することのできない SQL ファイルを開く場合使用するエンコードを選択することができます。

メモ エンコード指定のない SQL ファイルは、UTF-8 エンコードで保存されます。

SQL エディターの一般的な設定

SQL エディターに適用することのできる全般的な設定を以下のように変更することができます：

1. **「DB クエリ」** タブをクリックします。
2. ペインの上にある、**「オプション」** をクリックして、**「SQL エディター」** をクリックします。



全般

構文の色分けを有効化 により、SQL 構文における要素が異なる色により表示されるようになります。

タイムアウトダイアログを表示 チェックボックスにより、SQL ファイルが実行される際ならびにデータソースが接続されていない場合、対応するデータソースへの接続が自動的に行われます。

取得

SQL の実行が許可される最大時間（実行タイムアウト）を秒単位で定義します。

タイムアウトダイアログを表示 チェックボックスを有効にすることで、タイムアウトが発生した際にタイムアウトの設定を変更することができるようになります。

入力ヘルパーバッファー

SQL ステートメントの入力中の自動補完の提案を有効化するには、**「自動的に開く」** チェックボックスを選択します（次も参照する：自動補完）。

自動補完のための入力ヘルパーバッファーをデータソースに接続する際、または、最初に使用する際に入力することができます。バッファーの入力には時間がかかる可能性があります。**「バッファーのクリア」** ボタンを使用して、バッファーをリセットすることができます。

テキストビュー設定

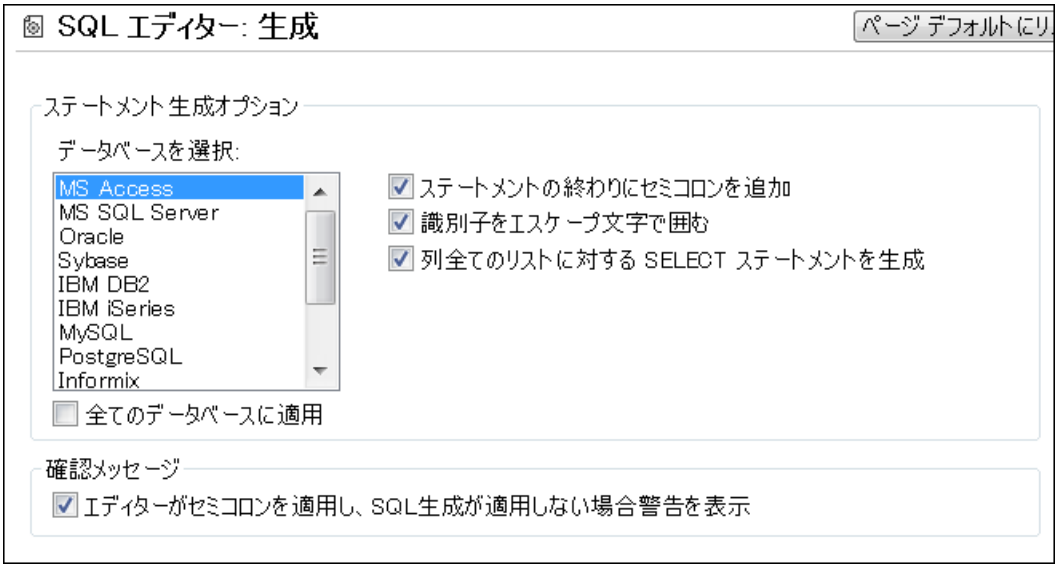
余白やタブ、視覚的な補助やテキストビューナビゲーションのホットキーといったテキストビューの設定を定義することが

できます。

SQL ステートメント生成の設定

SQL ステートメントの生成構文を複数のデータベースの型のために指定することができます：

1. **「DB クエリ」** タブをクリックします。
2. ペインの上にある、**「オプション」** をクリックして、**「SQL エディター」** をクリックします。



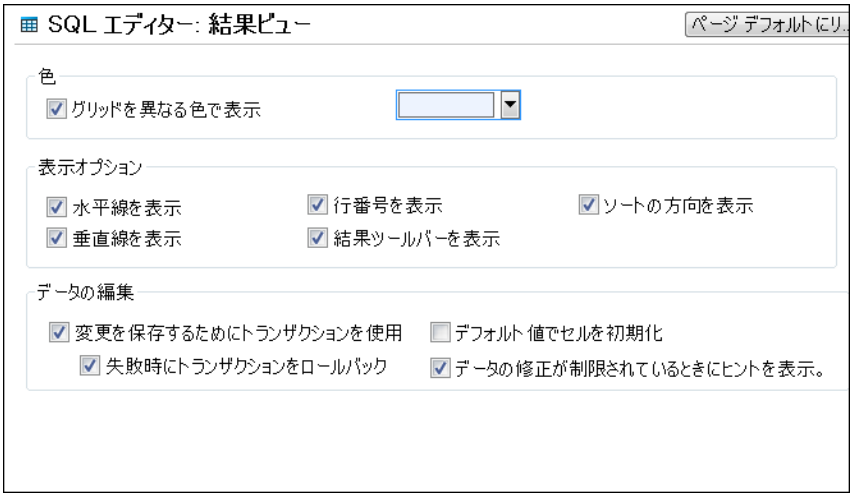
特定のデータベースのための構文の優先順序を定義するには、リストから選択して、右の3つのチェックボックスを有効化または無効化します。

すべての データベースのための一意の構文を定義するには、**全てのデータベースに適用**を選択して、右の3つのチェックボックスを有効化または無効化します。全てのデータベースのための共通の設定は、JDBC 接続を使用したOracle と IBM DB2 とiSeries データベース内のデータを編集する際、不都合が生じる可能性があることに注意してください。

クエリの結果ビューの設定

DB クエリペインの**結果**タブの外観の構成の方法 は、以下の通りです：

1. **DB クエリ** タブをクリックします。
2. ペインの上の、**オプション** をクリックし、**「SQL エディター | 結果ビュー」** をクリックします。



色を変更してグリッドを表示するを選択して結果タブを単純なグリッド表示にするか、または色を使った行の表示を行うか指定することができます。

表示オプショングループでは水平ならびに垂直のグリッド線、行番号や**結果**ツールバーなどの表示を指定することができます。対応するチェックボックスを有効/無効にすることで、これらの表示/非表示を切り替えることができます。

データの編集グループでは、トランザクションに関する設定や、デフォルトの値でセルを埋めるのか、またヒントの表示を行うかといった設定を行うことができます。

SQL エディターフォントの設定

SQL エディターに表示されるSQL ステートメントの色とフォントの設定を構成する方法は、以下の通りです：

1. **「DB クエリ」**タブをクリックします。
2. ペインの上にある、**「オプション」** をクリックして、**「SQL エディター | フォント」**をクリックします。



フォント設定リストボックスに表示されているフォント設定はSQL ステートメントの要素になります。SQL エディターに表示されるフォントフェイスやスタイル、サイズ、また色の指定を行うことができます。同一のフォントサイズが全てのテキストに対して使用されます。

個々のテキストタイプごとに指定することのできる設定はスタイルだけです。この機能により構文による色分けが可能になります。**ページデフォルトにリセット**ボタンをクリックすることで、オリジナルの設定に復元することができます。

7.2.10 ストアドプロシージャ

ストアドプロシージャとはデータベースサーバー上で動作するプログラムのことで、クライアントアプリケーションから呼び出されることになります。通常これらのプログラムは SQL を拡張した方言により記述され、データベースによっては、Java や .NET CLR といったプログラミング言語による実装をサポートしているものもあります。

通常ストアドプロシージャはデータベースの問い合わせのほか、検証済みの入力パラメーターによるデータベースへの修正などに使用されます。ストアドプロシージャは、電子メールの送信といったデータベースとは直接関係しない用途にも使用することができます。

MapForce におけるストアドプロシージャは以下のような特徴を持っています：

- ソースならびにターゲットデータベースコンポーネントとして表示、呼び出すことができます。
- 入力ならびに出力パラメーターによりデータをマッピングすることができます。
- 関数呼び出しのように挿入することができます。これにより入力データを与え、ストアドプロシージャにて実行された結果を出力データとして他のコンポーネントへ与えることができます。
- データベースがマッピング領域に挿入されると、ユニークな名前ならびにクリック可能なボタンがデータベースコンポーネント内部に表示されます。
- MapForce 内部にてストアドプロシージャを編集することはできません。
- 内蔵の実行エンジンでしか使用することはできず、C++、C#、Java によるコード生成ではサポートされません。

メモ： 以下にあるストアドプロシージャセクションでは、MS SQL 2008 とCodePlex Web サイト (http://sqlserversamples.codeplex.com) からダウンロードすることができるAdventureWorks データベースにより、MapForce へストアドプロシージャを使用する方法が示されます。

ストアドプロシージャのサポート：

- 入力/出力パラメーターの型：ユーザー定義型、カーソルの型、変数型、そしてデータベース固有のデータ型（配列、座標、CLR 型など）はサポートされていません。
- データベースによっては入力パラメーターにデフォルト値を定義することもできますが、この機能は現在サポートされていません。マッピングにて入力パラメーターへの接続が行われなかったとしても、デフォルト値が使用されることはありません。
- 複数のレコードを返すストアドプロシージャは、ドライバーとデータベースAPI (ODBC/ADO/JDBC) の組み合わせによってはサポートされます。固定されたカラム構造を持つ同じ数のレコードセットを返すプロシージャだけがサポートされます。
- Microsoft Access データベースのストアドプロシージャは機能に制限がありMapForceでサポートされていません。

ドライバー

- 通常、Microsoft から提供されているODBC や ADO、またはOracle から提供されているODBC や JDBC といったブリッジドライバーを使用するのは推奨されていません。できるかぎりデータベースベンダーから提供されている最新のネイティブデータベースドライバーを使用するようにしてください。

Firebird 2.5

- MapForce にてサポートされる対象：ストアドプロシージャ、テーブル値関数

IBM DB2 8, 9, 10, IBM DB2 for i 6.1, 7.1

- MapForce にてサポートされる対象：ストアドプロシージャ、スカラー関数、テーブル値関数。
- row-valued 関数（戻り値：行）はサポートされません。
- 実行後にエラー警告を読み取ることで発生する既知のJDBC ドライバー問題を回避するために、"IBM_DB2 9.7 Fix Pack 3a" の使用が推奨されます。これにより、存在しない結果セット行を作成するADO プロバイダーの問題も回避することができます。

Informix 11.7

- MapForce にてサポートされる対象：ストアドプロシージャ、テーブル値関数。

Microsoft SQL Server 2005, 2008, 2012, 2014

- MapForce にてサポートされる対象：ストアドプロシージャ、スカラー関数、テーブル値関数。
- **Microsoft OLE DB Provider for SQL Server** の代わりに最新の**SQL Server Native Client** ドライバーを使用することが奨励されます。
- SQL Server 2008 にて導入されたいくつかのデータ型（datatime2、datetimeoffset）に対して、ADO API は限られたサポートしか提供していません。ADO とSQL Server ネイティブクライアントを使用するなかで、新たな期間型にてデータの切り捨て問題に直面した場合、接続文字列にDataTypeCompatibility=80 という値を加えるか、ODBC を使用するようにしてください。

MySQL 5.5, 5.6

- MapForce にてサポートされる対象：ストアドプロシージャ、スカラー関数
- MySQL バージョン 5.5 .以降は、ストアドプロシージャに対する完全なサポートを含みます。これ以前のバージョンを使用する場合、MapForce の機能性は制限されます。

Oracle 9i, 10g, 11g, 12c

- MapForce にてサポートされる対象：ストアドプロシージャ、スカラー関数、テーブル値関数
- **Microsoft OLE DB Provider for Oracle** は非常に古いドライバーで、サポートされません。Oracle から提供される最新のドライバーを使用してください。
- Oracle ではREF CURSOR 型の出力パラメーターを使用することで結果セットが返されます。この仕様はストアドプロシージャに対してMapForce でサポートされるものの、関数に対してはサポートされません。このため、Oracle ストアドプロシージャに対して常に固定された数と名前が使用されることになります。

PostgreSQL 8, 9

- MapForce にてサポートされる対象：スカラー関数、row-valued 関数、テーブル値関数。
- PostgreSQL では、関数にて定義された任意の出力パラメーターにより、結果セットのカラムが記述されます。この情報はMapForce にて自動的に使用され、情報取得のための実行やレコードセットの入力を手動で行う必要はありません。refcursor 型のパラメーターはサポートされていません。

Progress OpenEdge 11.6

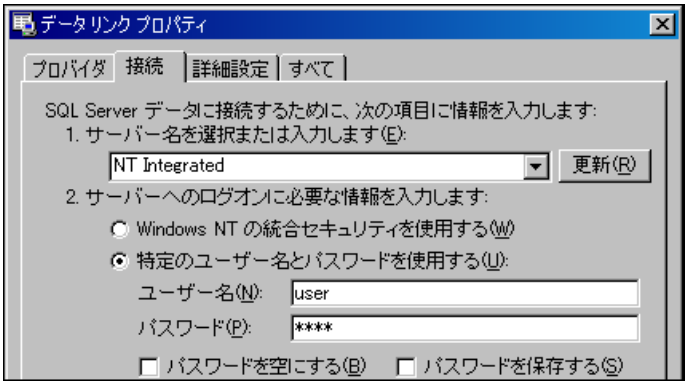
- MapForce にてサポートされる対象：ストアドプロシージャ

7.2.10.1 データベースコンポーネントへストアドプロシージャを挿入

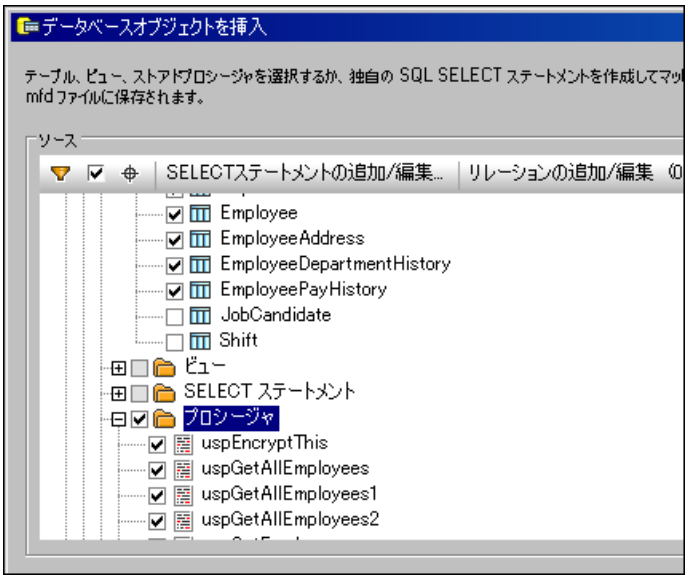
ストアドプロシージャをマッピング領域へ挿入することで、データベースコンポーネントとともに使用することができるようになります。挿入された後は、通常のデータベースコンポーネントと同じように使用することができます。

ストアドプロシージャを含むデータベースコンポーネントを挿入する：

1. 「データベースを挿入」アイコンをクリックするか、メニューオプションから **「ファイル| データベースを挿入」** を選択します。
2. 「接続ウィザード」によりデータベースへ接続します。
3. データリンクプロパティダイアログボックスの接続タブにて必要な情報を入力し、**「OK」**ボタンをクリックします。



4. 展開アイコンをクリックして、挿入を行うデータベーステーブルを選択します。

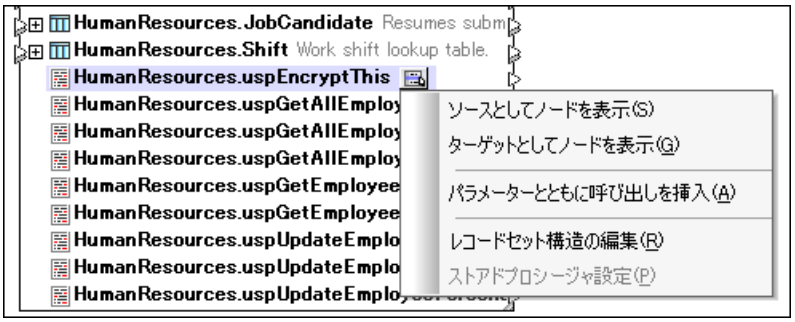


5. プロシージャフォルダーの展開アイコンをクリックして、テーブルとともに挿入するストアドプロシージャを選択し、**「OK」**をクリックします。



データベースコンポーネントが挿入され、選択されたテーブルとストアドプロシージャが表示されます。

- テーブル、ビュー、そしてストアドプロシージャがアルファベット順にてデータベースコンポーネント内部へ配置されます。
- データベースコンポーネント内のアイテムとして表示される各ストアドプロシージャには、プロシージャ名とクリックすることのできるボタンが表示されます。このボタンにより、ストアドプロシージャをソースまたはターゲットとして使用するのかなどの設定を指定することができます。
- この段階ではストアドプロシージャのパラメーターがソースパラメーターとして使用されるのか、ターゲットパラメーターとして使用されるのかの情報がMapForce へ与えられていません。ストアドプロシージャボタンをクリックして、適切なオプションを選択する必要があります。



7.2.10.2 使用目的

以下のテーブルにより、ストアドプロシージャの殆どの種類とMapForce における定義方法がカバーされます。

使用目的：	参照すべきセクション
ストアドプロシージャを呼び出すことで、データベースからデータを**抽出**して、他のコンポーネントへのマッピングを行う。 例：ストアドプロシージャをデータソースとして使用して、その結果を他のファイル（XML、TXT、EDI など）へ書き込む。	ソースコンポーネント内のストアドプロシージャ
ストアドプロシージャを呼び出すことで、データベースの**修正**やその他のアクションを行う。	ターゲットコンポーネント内のストアドプロシージャ
ストアドプロシージャを使用して、INSERT ステートメントにて使用される値（キー）を**生成**する。	主キーを生成するためにストアドプロシージャを使用する

7.2.10.3 ストアドプロシージャとローカルリレーション

ローカルリレーションを使用することで、ストアドプロシージャの呼び出しや（INSERT や UPDATE といった）データベーステーブルにおけるアクションの実行を行う際の階層的な順序を定義することができます。この機能はソースならびにターゲットコンポーネントにて使用することができます（次を参照：ローカルリレーションシップの定義）。

ローカルリレーションには（主キー／一意キーを含む）親オブジェクトと（外部キーを含む）子オブジェクトが常に与えられます。

リレーションで使用することのできる親オブジェクトとそのフィールド：

- データベーステーブルまたはビュー（カラム）
- ストアドプロシージャ（出力パラメーターまたは戻り値）
- ストアドプロシージャのレコードセット（カラム） - ソースとプロシージャの呼び出しコンポーネントに対してのみ使用でき

ます
- ユーザー定義 SELECT ステートメント (カラム)

使用することのできる子オブジェクトとそのフィールド：:
- データベーステーブルまたはビュー (カラム、WHERE 条件を生成)
- ストアドプロシージャ (入力パラメーター)
- ユーザー定義 SELECT ステートメント (入力パラメーター)

ソースコンポーネントでは、これにより関連するオブジェクトからデータを読み取るのが簡単になります。例えばデータベーステーブルからID を読み取り、これらID によりストアドプロシージャを呼び出し、関連する情報を抽出することができます。別のプロシージャから取得したデータとともにストアドプロシージャを呼び出すこともできます。

ターゲットコンポーネントでは、あるプロシージャによりID 値を作成し、別のプロシージャにより関連する情報が別のテーブルに挿入されるといった、複数のプロシージャを呼び出す順序を定義することができます。ローカルリレーションのストアドプロシージャとテーブルを混ぜ合わせることも可能で、別のプロシージャを呼び出す代わりに関連するテーブルに直接挿入アクションを行うということができます。

7.2.10.4 データソースとしてのストアドプロシージャ

ストアドプロシージャの出力は 0個以上の出力や戻り値パラメーターのほか、ストアドプロシージャ内部にて記述された SELECT ステートメントから得られるレコードセットとなります。ストアドプロシージャ内部にあるSELECT ステートメントから得られるレコードセットはテーブルやビューと同等に扱うことができます。出力パラメーターとレコードセットをターゲットコンポーネントへマッピングすることができます。

これらレコードセットのカラム構造はデータベースカタログから直接読み取ることができないため、デザイン時にストアドプロシージャを実行して検知するか、手動で定義する必要があります。詳細については レコードセットの定義 を参照ください。

ストアドプロシージャに入力パラメーターが与えられているかにより、MapForce におけるストアドプロシージャの扱いは変化します：

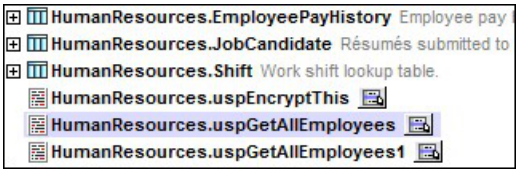
ストアドプロシージャに入力パラメーターが与えられていない	入力パラメーターを持たないストアドプロシージャ
XML、テキスト、その他のファイルや定数などにより、ストアドプロシージャの入力パラメーターへ値を与える	パラメーターを伴う呼び出し -入力と出力
同じデータベースにあるテーブルやビュー、または別のストアドプロシージャから得られた値をストアドプロシージャの入力パラメーターとして与える	ソースコンポーネントとローカルリレーション

入力パラメーターを持たないストアドプロシージャ

入力パラメーターを持たないソースコンポーネントとしてストアドプロシージャを使用する場合に、このオプションを使用することができます。

例 :純粋な SELECT ステートメントとして機能するするストアドプロシージャに入力パラメーターを与える必要はなく、SELECT ステートメントの結果をターゲットコンポーネントへマッピングすることができます。

AdventureWorks データベースの HumanResources.uspGetAllEmployees を例に考えてみましょう。



ストアドプロシージャ:

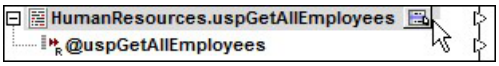
```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

MapForce ではレコードセットのカラムをデータベースカタログから直接読み取ることができないため、デザイン時にストアドプロシージャを実行して検知するか、手動で定義する必要があります。

ソースコンポーネントの出力レコードセットを定義する:

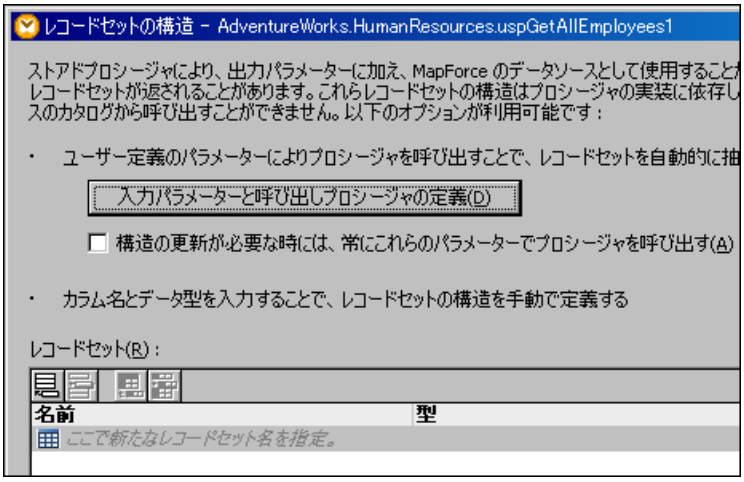
AdventureWorks データベースコンポーネントを挿入し、ストアドプロシージャとともにHumanResources テーブルを選択します::

1. ストアドプロシージャボタンをクリックし、ソースとしてノードを表示オプションを選択します。

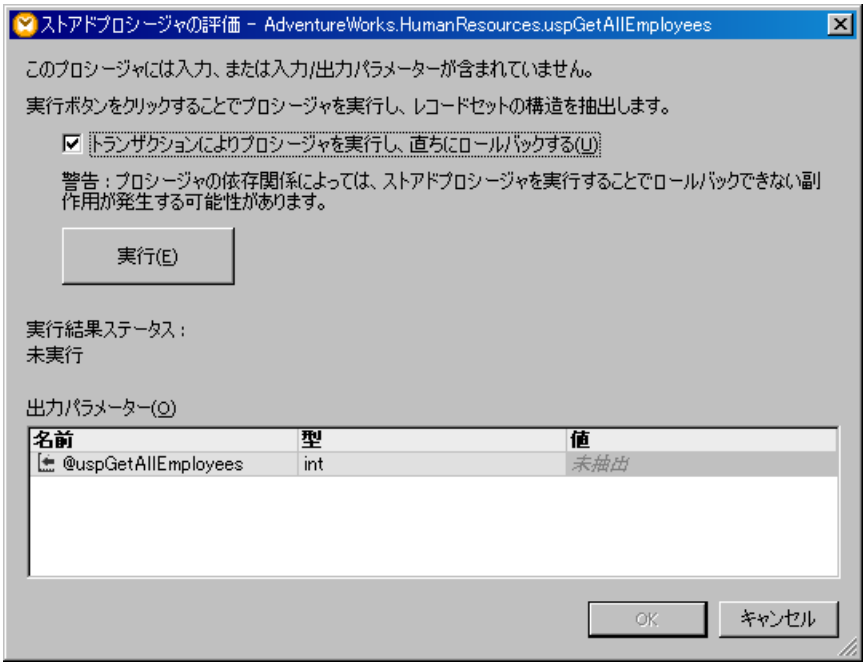


戻り値ノードがストアドプロシージャ名の下に挿入されます。必要なのは戻り値ではなくレコードセットなので、ストアドプロシージャボタンを再度クリックして、**「レコードセット構造の編集」**を選択します。

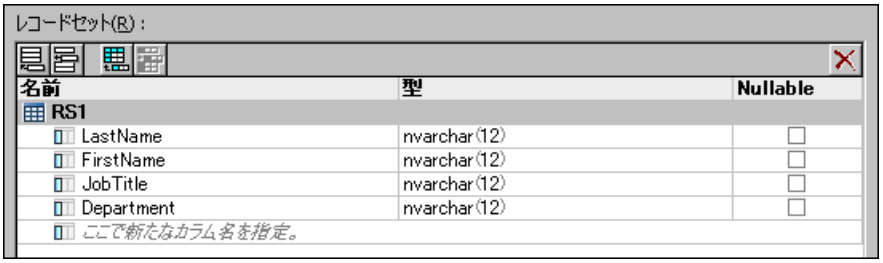
2. 入力パラメーターと呼び出しプロシージャの定義ボタンをクリックします。



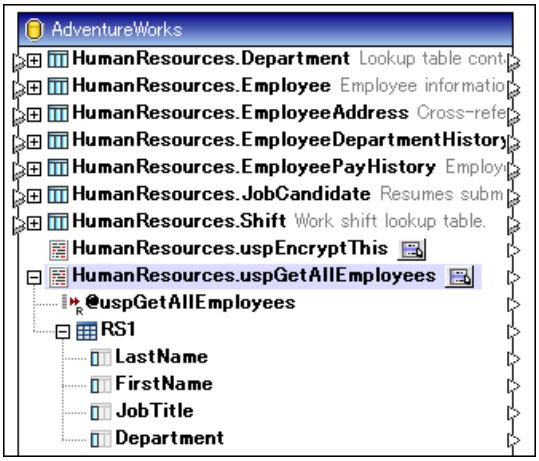
ストアドプロシージャの評価ダイアログボックスが表示されます。



3. 実行ボタンをクリックして、「OK」をクリックします。
ダイアログボックスのレコードセットセクションにてレコードセットが表示されます。

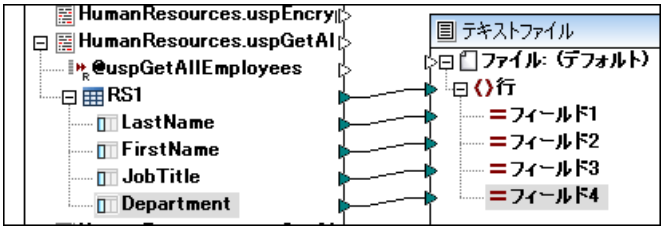


4. 「OK」ボタンを再度クリックして、レコードセットの定義を完了します。
LastName といった名前のカラムがレコードセットノード RS1 の下に表示されます（レコードセットが表示されていない場合、＋ボタンをクリックしてください）。



マッピングを完了する：

1. テキストファイルコンポーネントを挿入して、出力アイコンをテキストファイルへマッピングします。



2. 出力ボタンをクリックして結果を確認します。

```
1    Gilbert,Guy,Production Technician - WC60,Production
2    Brown,Kevin,Marketing Assistant,Marketing
3    Tamburello,Roberto,Engineering Manager,Engineering
4    Walters,Rob,Senior Tool Designer,Tool Design
5    D'Hers,Thierry,Tool Designer,Tool Design
6    Bradley,David,Marketing Manager,Marketing
7    Dobney,JoLynn,Production Supervisor - WC60,Production
```

メモ:
デザイン時にストアドプロシージャを実行するにあたり、回避するべき副作用が発生する場合、レコードセットの構造ダイアログボックスにて手動でレコードセットを定義することもできます。レコードセットの構造ダイアログボックスにてレコードセットの追加またはカラムの追加ボタンをクリックしてください。

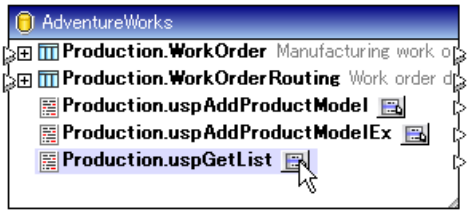
パラメーターを伴う呼び出し - 入力と出力

ストアドプロシージャは関数呼び出しのように使用することもできます。これにより、以下の処理を行うことができます:

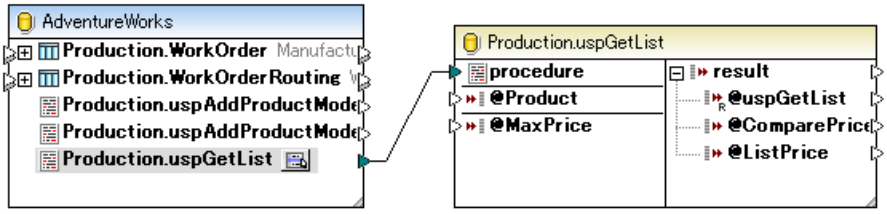
- 入力データをプロシージャに与える
- プロシージャを実行する
- プロシージャの出力データを他のコンポーネントへマッピングする

ストアドプロシージャを関数呼び出しのように使用する:
AdventureWorks データベースコンポーネントを挿入 し、ストアドプロシージャとともにProduction テーブルを選択します。

1. Production.uspGetList のストアドプロシージャボタンをクリックして、メニューを開きます。



2. パラメーターとともに呼び出しを挿入オプションを選択します。

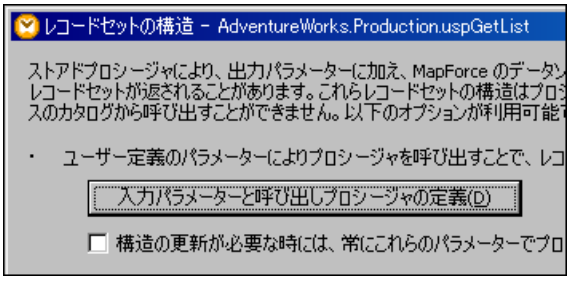


この操作によりプロシージャコンポーネントがマッピングに挿入されます。このコンポーネントはWeb サービスやユーザー定義コンポーネントと同様に動作します。プロシージャ名がコンポーネントのprocedure アイテムへ自動的に接続されます。

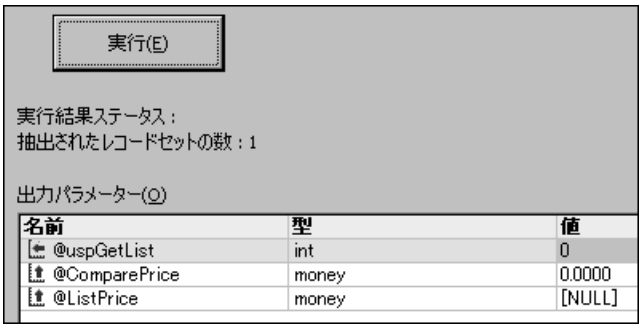
コンポーネントの左側にはprocedure 入力パラメーターが表示され、右側には出力パラメーターが表示されます。このストアドプロシージャの場合、出力パラメーターとレコードセットが返されますが、それらをMapForce で使用する前に、構造を定義する必要があります。

レコードセットの構造を定義する：

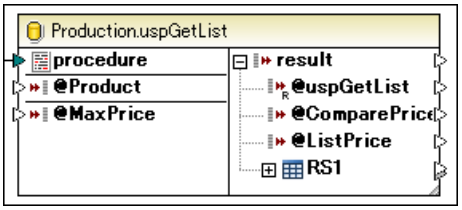
1. uspGetList のストアドプロシージャボタンをクリックして、レコードセット構造の編集を選択します。
2. 入力パラメーターと呼び出しプロシージャの定義ボタンをクリックして、表示されるダイアログボックスにて実行ボタンをクリックします。



この操作により、返された出力パラメーター値がダイアログボックス下部にあるテーブルに表示され、レコードセットが 1 つ抽出されたことが表示されます。



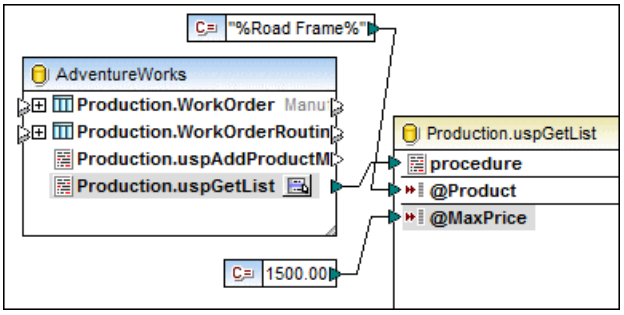
3. 「OK」ボタンをクリックして確定し、再度 「OK」ボタンをクリックすることでレコードセットダイアログボックスを閉じます。



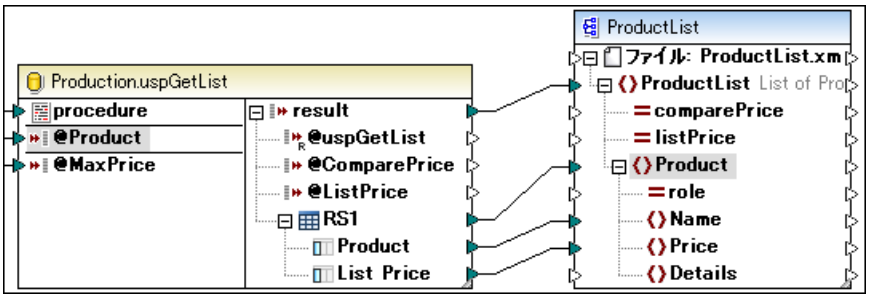
レコードセットがストアドプロシージャコンポーネントの出力セクションに追加されました。

呼び出しパラメーターコンポーネントを使用する：

1. 入力パラメーターを与えるために使用するコンポーネントを定義します。例えば、以下のスクリーンショットでは、 2つの定数コンポーネントが定義されており、入力パラメーターへ接続されています。



2. ストアドプロシージャの出力を含むことになるターゲットコンポーネントを定義します。以下の例ではXML ドキュメントが使用されています。



3. 出力ボタンをクリックすることで、マッピングの結果を確認することができます。

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <ProductList xsi:noNamespaceSchemaLocation="C:/Tools/sp/ProductList.xsd"
3     <Product>
4       <Name>HL Road Frame - Black, 58</Name>
5       <Price>1431.5</Price>
6     </Product>
7     <Product>
8       <Name>HL Road Frame - Red, 58</Name>
9       <Price>1431.5</Price>
10    </Product>
11    <Product>
12      <Name>HL Road Frame - Red, 62</Name>
13      <Price>1431.5</Price>
14    </Product>
15    <Product>
16      <Name>HL Road Frame - Red, 44</Name>
17      <Price>1431.5</Price>
```

この例における出力では各種製品が表示されています。

ソースコンポーネントとローカル リレーション

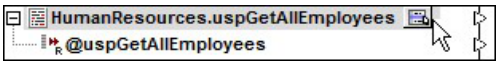
このオプションを使用することで、ストアドプロシージャから得られたレコードセットに、直接関係の無い他のテーブルから得られたデータを組み合わせることができます。

```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

MapForce はレコードセットのカラムをデータベースのカタログから直接読み取ることができないため、ストアドプロシージャを一度実行してカラムを検出するか、手動で定義する必要があります。

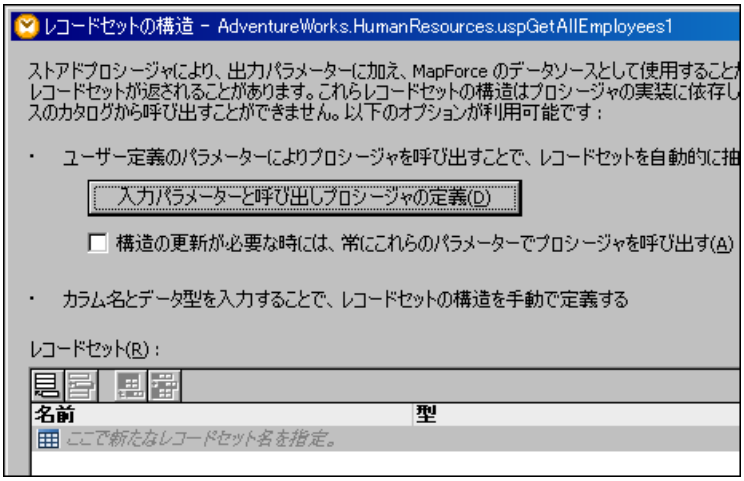
ソースコンポーネントの出力レコードセットを定義する：
AdventureWorks データベースコンポーネントを挿入 し、ストアドプロシージャとともにHumanResources テーブルを選択します

1. ストアドプロシージャボタンをクリックして、ソースとしてノードを表示オプションを選択します。

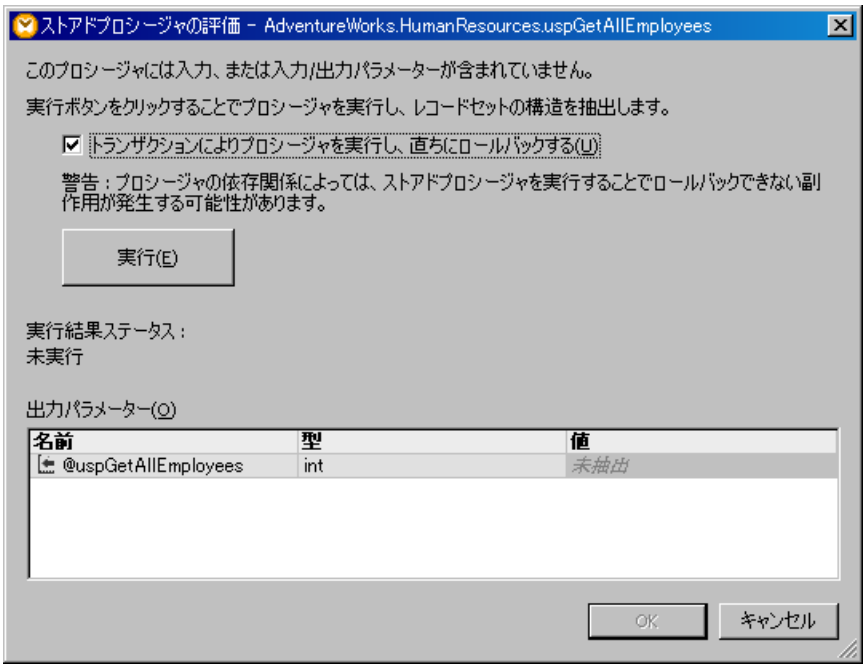


戻り値ノードがストアドプロシージャ名以下に挿入されます。必要としているのは戻り値の値ではなくレコードセットなので、ストアドプロシージャボタンを再度クリックして、 **「レコードセット構造の編集」**オプションを選択します。

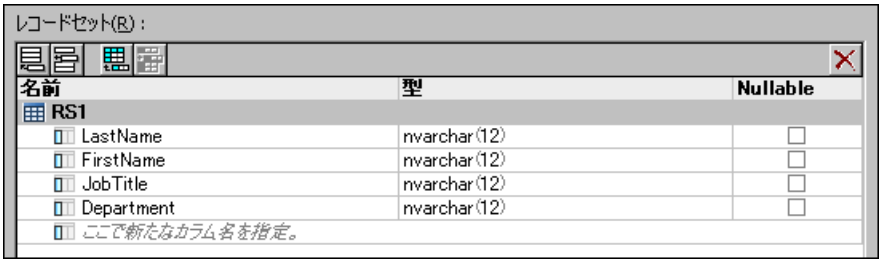
2. 入力パラメーターと呼び出しプロシージャの定義ボタンをクリックします。



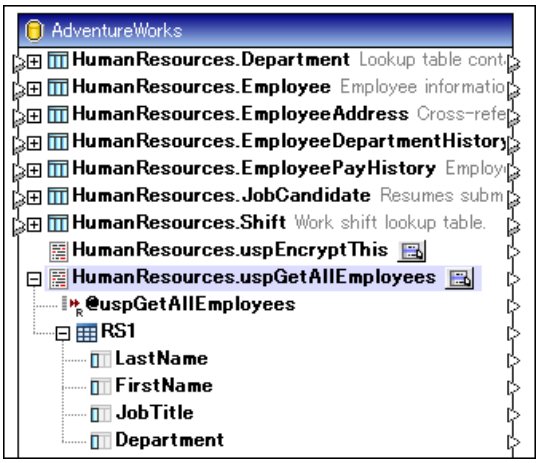
ストアドプロシージャの評価ダイアログボックスが表示されます。



3. 実行ボタンをクリックし、 **「OK」**をクリックします。
ダイアログボックスのレコードセットセクションにてレコードセットのフィールドが表示されます。

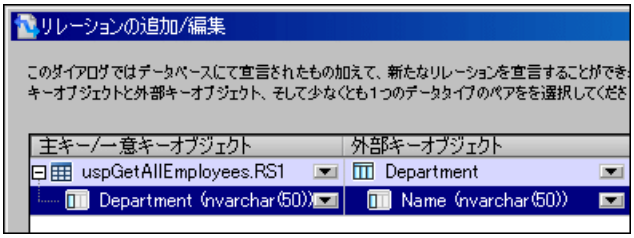


4. 「OK」ボタンをクリックして、レコードセットの定義を完了します。
レコードセットノードのRS1 以下にLastName などのカラムが表示されます (表示されていない場合、 + アイコンにより展開してください)。

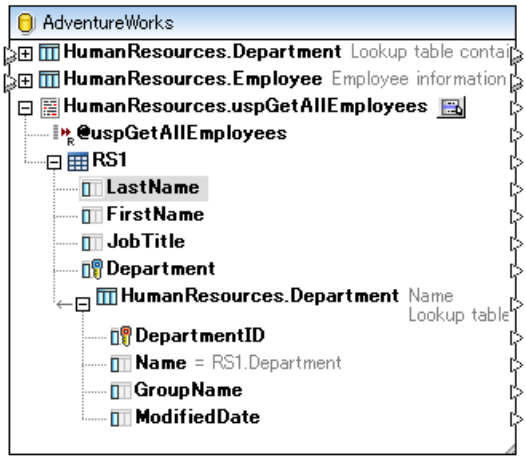


別のテーブルに対するローカルリレーションを定義する:

1. コンポーネントヘッダーを右クリックして、データベースオブジェクトの追加 削除 編集を選択します。
2. 表示されるダイアログにてリレーションの追加 編集ボタンをクリックして、リレーションの追加 編集ダイアログを表示し、リレーションを追加ボタンをクリックします。
3. ストアドプロシージャの**uspGetAllEmployees.RS1** を主キー/一意キーオブジェクトとして、**@Department** パラメーターをカラムとして定義します。
4. **Department** を外部キーとして、**Name** をカラムとして定義します。



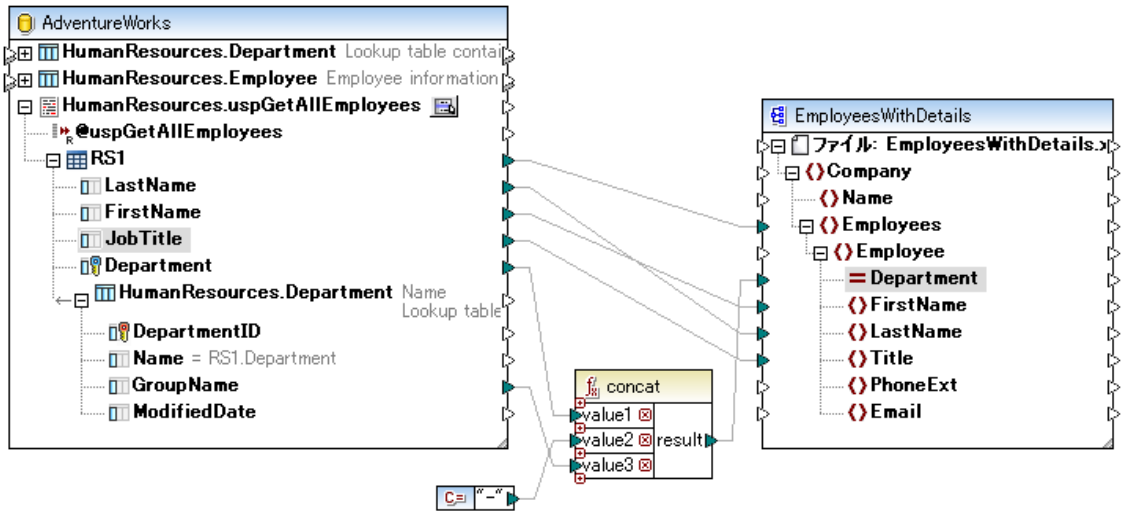
5. ダイアログボックス内の 「OK」ボタンをクリックします。



Department テーブルがストアドプロシージャの子として表示されます。

マッピングを完成させる：

1. ソースデータベースのデータをマッピングするターゲットスキーマを挿入し、以下に示されるような接続を追加します。



2. 出力ボタンをクリックすることで、マッピングの結果を確認することができます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Company xsi:noNamespaceSchemaLocation="C:/Users/Documents/Altova/Ma
  <Employees>
    <Employee Department="Production - Manufacturing">
      <FirstName>Guy</FirstName>
      <LastName>Gilbert</LastName>
      <Title>Production Technician - WC60</Title>
    </Employee>
  </Employees>
  <Employees>
    <Employee Department="Marketing - Sales and Marketing">
      <FirstName>Kevin</FirstName>
      <LastName>Brown</LastName>
      <Title>Marketing Assistant</Title>
    </Employee>
  </Employees>
  <Employees>
    <Employee Department="Engineering - Research and Development">
      <FirstName>Roberto</FirstName>
      <LastName>Tamburello</LastName>
      <Title>Engineering Manager</Title>
    </Employee>
```

7.2.10.5 ターゲットコンポーネントにおけるストアドプロシージャ

ストアドプロシージャを使用することで (ADD / UPDATE / DELETE などにより) データベースへ変更を加え、ストアドプロシージャからの出力を必要としない場合には、このオプションを選択してください。

ターゲットコンポーネントにてストアドプロシージャを使用する:
このオプションによりターゲットデータベースコンポーネント内にあるストアドプロシージャアイテム以下の**入力**パラメーターに、子ノードが追加されます。

AdventureWorks データベースの uspAddProductModel ストアドプロシージャを使用することで、新たなモデルをデータベースへ追加する例を考えてみましょう。



ストアド プロシージャ:

```
PROCEDURE Production.uspAddProductModel
@ModelName nvarchar(50),
@Inst xml

as
INSERT INTO [AdventureWorks].[Production].[ProductModel]
           ([Name]
           --,[CatalogDescription]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
           --,<CatalogDescription, ProductDescriptionSchemaCollection,>
           ,@Inst
           ,NEWID()
           ,GETDATE());
```

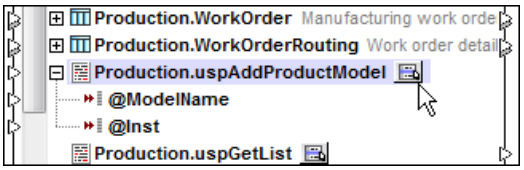
ランタイムでは、マッピングされた入力パラメーターとともにストアドプロシージャが実行され、そのデータ出力は無視されることに

なります。

入力パラメーターアイテムをターゲットコンポーネントにて作成する：

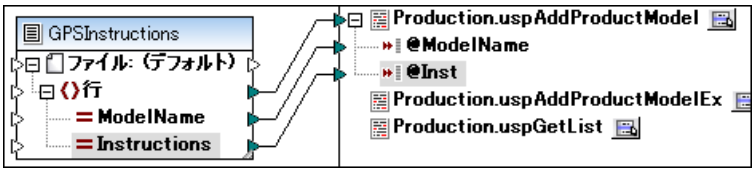
AdventureWorks データベースコンポーネントを挿入 し、ストアドプロシージャとともにProduction テーブルを選択します。

1. ストアドプロシージャボタンをクリックして、ターゲットとしてノードを表示オプションを選択します。



この操作により@ModelName ならびに@Inst 入力パラメーターがストアドプロシージャ名以下に表示されます。ターゲットコンポーネントに表示される入力パラメーターには入力アイコンだけが表示されます。

2. テキストファイル、XML ファイルといったソースコンポーネントを挿入し、入力パラメーターデータを提供するアイテムをストアドプロシージャの入力アイテムへマッピングします。



ストアドプロシージャに対するトランザクションを定義する：

1. ストアドプロシージャボタンをクリックして、ストアドプロシージャ設定オプションを選択します。
データベースストアドプロシージャ設定ダイアログボックスが表示されます。



2. トランザクションを使用チェックボックスにチェックを入れ、「OK」ボタンをクリックして確定します。
トランザクション設定により、実行中にエラーが発生した場合でも、プロシージャコマンドをロールバックできるようになります。
3. 出力ボタンをクリックして、データベースへ送信されるコマンドを確認することができます。

```
BEGIN TRANSACTION

EXECUTE NULL = [Production].[uspAddProductModel] 'GPS Navigator',' GPS instructions';

COMMIT TRANSACTION
```

このダイアログボックスでは、ストアドプロシージャが呼び出される前に実行されるSQL ステートメントを定義することもできます。

メモ:
各パラメーターはアトミック値であるため (かつ "nullable" とすることもできるため)、ストアドプロシージャの**パラメーター**に対して、入力を複製して追加コンテキストメニューオプションを選択することはできません。

入力を複製して追加コンテキストメニューオプションはストアド**プロシージャ**アイテムに対して使用することができます。ストアドプロシージャを複製することで、それぞれのアイテム/ノードからストアドプロシージャが呼び出されることになります。

ストアドプロシージャを使用した主キーの生成

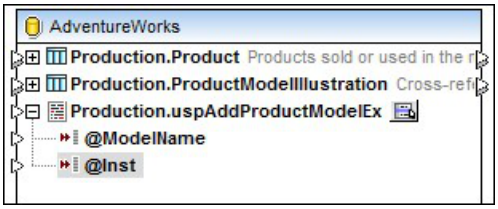
ストアドプロシージャによりデータベーステーブルへの変更が行われ、ストアドプロシージャから得られる出力パラメーターにより別のテーブルにおける主キーが作成されるような場合にこのオプションを選択してください。

AdventureWorks データベースにある uspAddProductModel の派生系である uspAddProductModelEx ストアドプロシージャを使用します

```
procedure Production.uspAddProductModelEx
@ModelName nvarchar(50),
@Inst xml,
@ProductModelID int OUTPUT
as begin
INSERT INTO [AdventureWorks].[Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
           ,@Inst
           ,NEWID()
           ,GETDATE());
   SELECT @ProductModelID = SCOPE_IDENTITY()
end;
```

AdventureWorks データベースコンポーネントを挿入 し、ストアドプロシージャとともに Production テーブルを選択します。

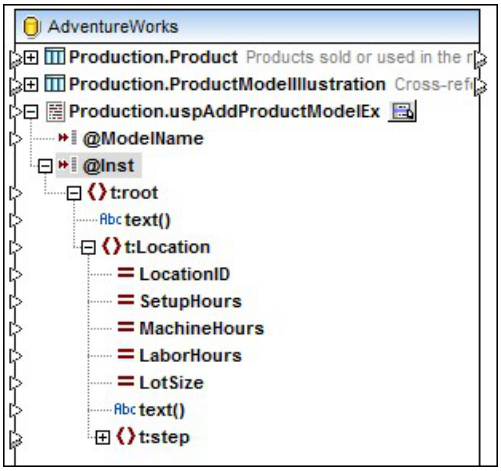
1. ストアドプロシージャボタンをクリックして、ターゲットとしてノードを表示オプションを選択します。



この操作により、プロシージャ名以下に ModelName と Inst パラメーターが挿入されます。コンポーネントにはストアドプロシージャの入力パラメーターだけが表示されます。

Inst パラメーターは XML 型であるため、対応する XML スキーマを割り当てることで、XML データを与えることができます。

2. Inst パラメーターを右クリックして、フィールドに XML スキーマを割り当てを選択します。
3. データベースコンボボックスにて Production.ManuInstructionsSchemaCollection を選択し、**「OK」**をクリックします。



この操作により、XML スキーマの要素と属性がコンポーネントへ追加されます。ModelName パラメーターとInst パラメーターをコンポーネントにて使用することができます。

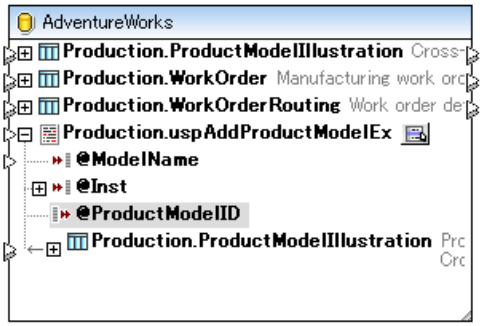
ここで、ストアドプロシージャパラメーター (production.product) により参照されているテーブルへ直接接続されていないテーブルに対してローカルリレーションを定義することにします。

主キーを生成するテーブルに対してローカルリレーションを定義する：

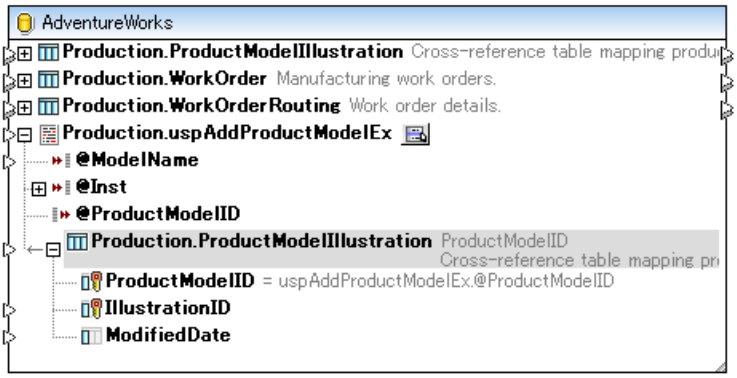
1. コンポーネントヘッダーを右クリックして、データベースオブジェクトの追加 削除 編集を選択します。
2. 表示されるダイアログにてリレーションの追加 編集ボタンをクリックして、リレーションの追加 編集ダイアログを表示し、リレーションを追加ボタンをクリックします。
3. 主キー／一意キーとしてストアドプロシージャの**uspAddProductModelEx** を定義し、カラムに **@ProductModelID** パラメーターをセットします。
4. 外部キーオブジェクトとして**ProductModelIllustration** を定義し、カラムには**ProductModelID** をセットします。



5. 「OK」ボタンを (数回) クリックします。



- ストアドプロシージャの出力パラメーター (ProductModelID) がストアドプロシージャへ追加されました。パラメーターにはローカルリレーションにて使用されるアイコンが表示されているものの、入力や出力アイコンは表示されていません。
- ストアドプロシージャの子アイテムとして、ProductModelIllustration テーブルが追加されました。
- テーブルを展開することで、テーブルのキーとカラムを確認することができます。ProductModelID キーには関連付けられたストアドプロシージャとパラメーターが表示されていることに注目してください。
- ストアドプロシージャの (出力) レコードセットを使用するローカルリレーションを、ここで使用することはできません。
- ストアドプロシージャボタンをクリックして、ストアドプロシージャ設定オプションを選択することで、プロシージャが呼び出される前に実行される SQL ステートメントを定義、またはトランザクション設定を有効にすることができます。



ストアドプロシージャに対してトランザクションを定義する：

1. ストアドプロシージャアイコンをクリックして、ストアドプロシージャ設定を選択します。
2. トランザクションを使用チェックボックスにチェックを入れ、「OK」をクリックして確定します。

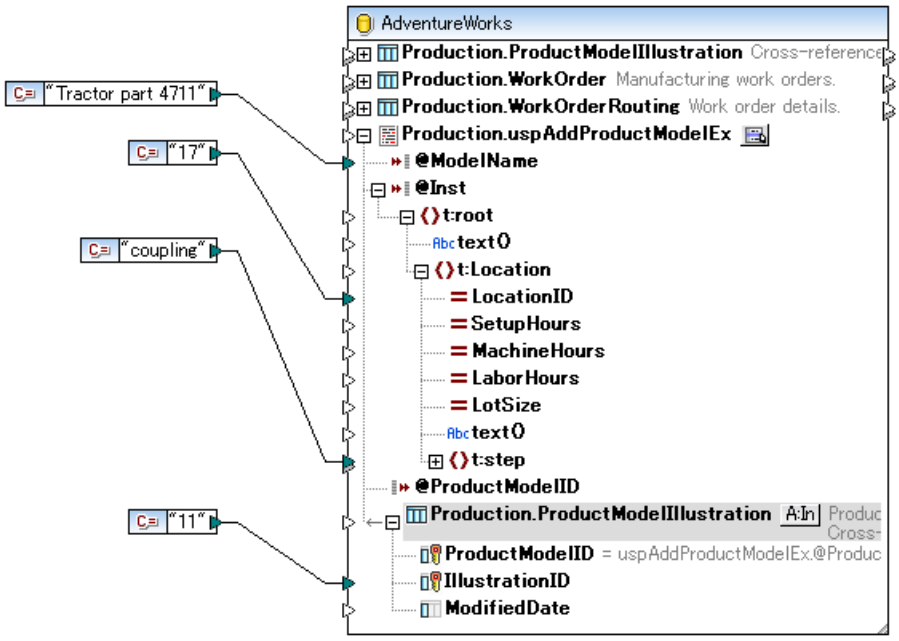


ストアドプロシージャに対してトランザクションを定義することにより、キーの抽出とレコードの挿入が同一のトランザク

ションで発生することが保証されます。

マッピングを完成させる：
通常行われるマッピングの一部を以下のスクリーンショットに示します。

1. データソースアイテム（この例では定数）をターゲットのデータベースへマッピングします。



2. 出力ボタンをクリックして、データベースへ送信される擬似 SQL を確認することができます。

```
BEGIN TRANSACTION

EXECUTE NULL = [Production].[uspAddProductModelEx] 'Tractor part 4711','<root
xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/ProductModelManuInstructions
"><Location LocationID="17"><step>coupling</step></Location></root>',NULL OUT;
-->>> %@uspAddProductModelEx1%
-->>> %@ProductModelID1%

INSERT INTO [Production].[ProductModelIllustration] ([ProductModelID], [IllustrationID]) VALUES (
'%@ProductModelID1%', 11)

COMMIT TRANSACTION
```

挿入アクションが行われる前に、各レコードに対してストアドプロシージャが自動的に呼び出されるようになります。

7.3 CSV とテキストファイル

MapForce は、CSV (コンマで区切られた値) とFLF (固定長フィールド) テキストファイルなどのテキストベースのファイルから、またはへの、データのマッピングのためのサポートを含みます。 例 :以下のようなデータ変換を作成することができます :

- XML スキーマとフラットファイルフォーマット
- データベースとフラットファイルフォーマット
- UN/EDIFACT ならびにANSI X12 と フラットファイルまたはデータベース

CSV の場合、ファイルはコンマだけでなく、タブ、セミコロン、スペースまたは他のカスタム化された値などの区切り記号を含むことができます。

CSV とFLF ファイルの他に、MapForce FlexText (このモジュールはMapForce Enterprise Edition で使用することができます) を使用すると、複雑またはカスタム化された構造のためのテキストファイルから、またはへのマッピングが可能になります。FlexText を使用すると、("FlexText テンプレート"を使用して)他のフォーマットにマッピングするために カスタムテキストデータを定義することが可能になります。詳細は MapForce FlexText を参照ください。

ファイルテキストから、または、テキストファイルへのデータのマッピングは以下の言語でサポートされています :Java、C#、C++、または BUILT-IN。

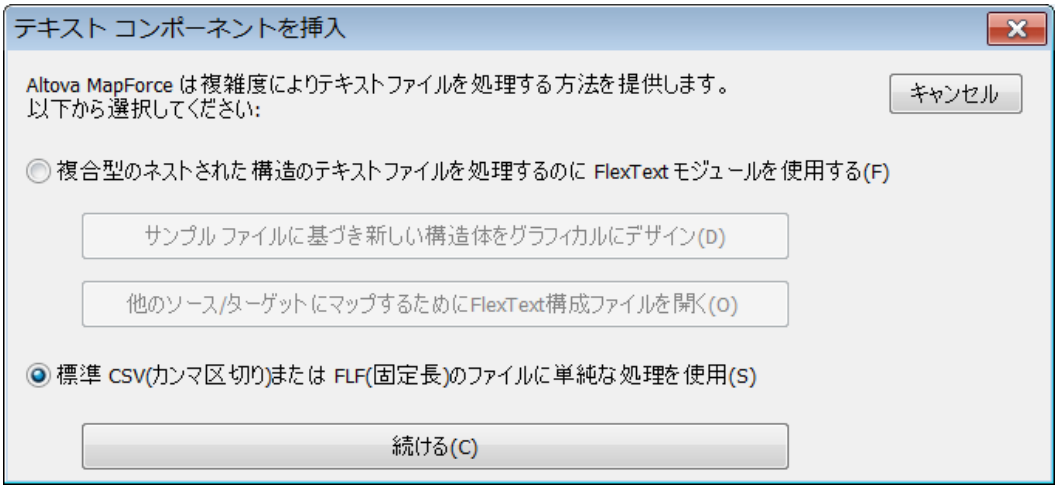
マッピングされたフラットなファイルデータ生成 /保存するには 2 つの方法があります :

- 出力タブをクリックすることで、内蔵の実行エンジンを使ったプレビューが生成され、「出力 | 出力ファイルの保存」メニューオプションを選択するか、 アイコンをクリックすることで結果を保存できます。
- メニューオプションの「ファイル | コードの生成 | Java、C#、またはC++」を選択し、生成されたコードをコンパイル、実行します。

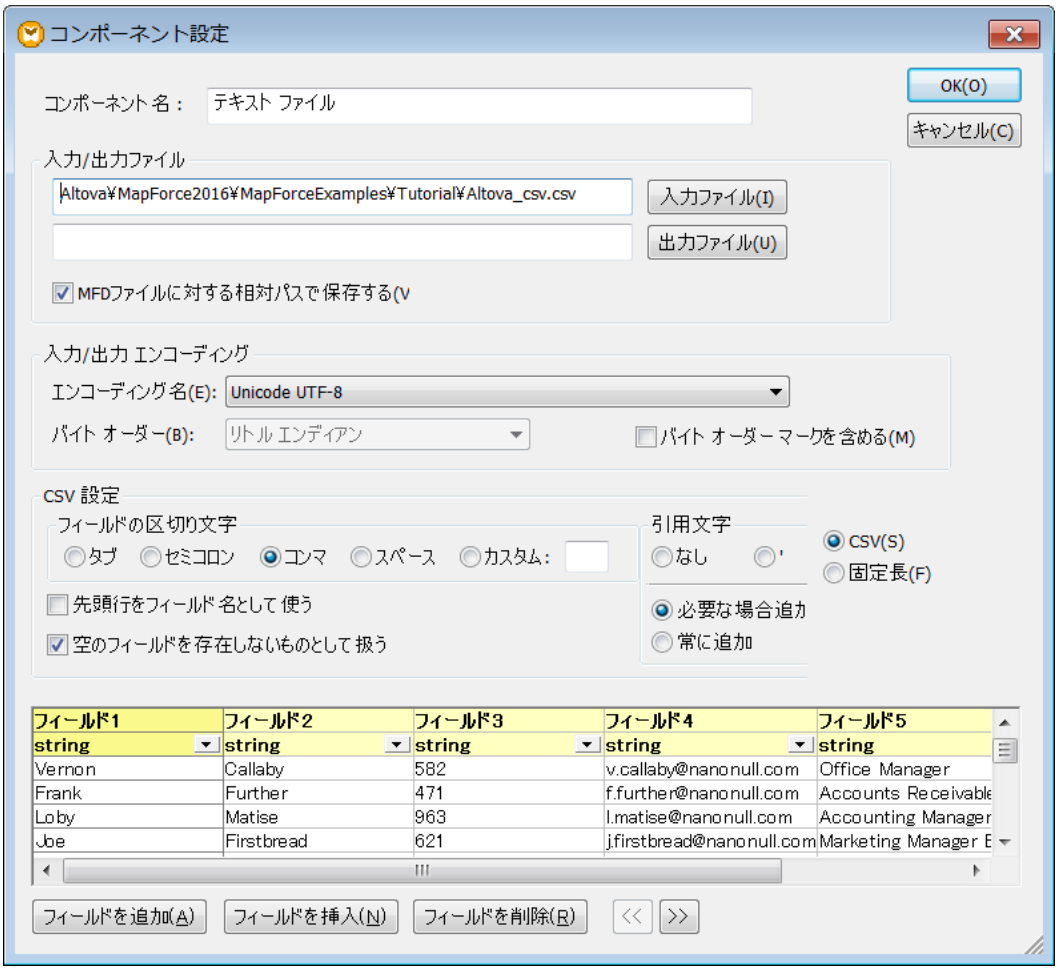
7.3.1 サンプル :CSV ファイルから XML へのマッピング

この例ではMFCompany.xsd スキーマファイルをベースに、単純なCSV ファイルをXML ファイルへマッピングします。以下の例で使用されるファイルは全て **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーに収められています。

1. ツールバーにてJava、C#、C++ またはBUILT-IN のアイコンが選択されていることを確認します。
2. メニューオプションの「挿入 | テキストファイル」を選択するか、テキストファイルを挿入アイコン ( ) をクリックします。「テキストコンポーネントを挿入」ダイアログボックスが表示されます。

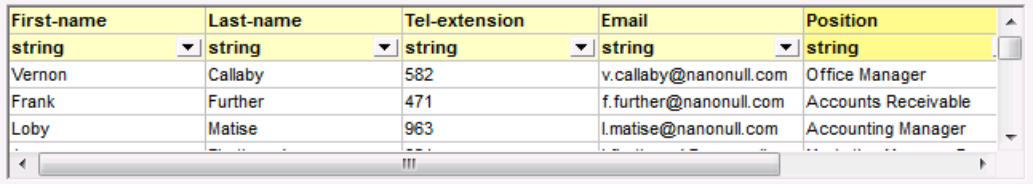


3. 標準 CSV または固定長のファイル...のラジオボタンをクリックして、続けるボタンをクリックします。
4. コンポーネント設定ダイアログボックスから、**入力ファイル**をクリックして、**Altova_csv.csv** ファイルを参照します。ファイルのコンテンツは、ダイアログボックスの下の部分に表示されます。プレビューモードの場合最初の20列のみが表示されることに注意してください。



5. **フィールド 1**のヘッダーをクリックして、テキストの内容を 例えばFirst-name に編集します。他のフィールドも同様の方法で編集します：フィールド 2 => Last-name、フィールド 3 =>Tel-extension、フィールド 4 =>

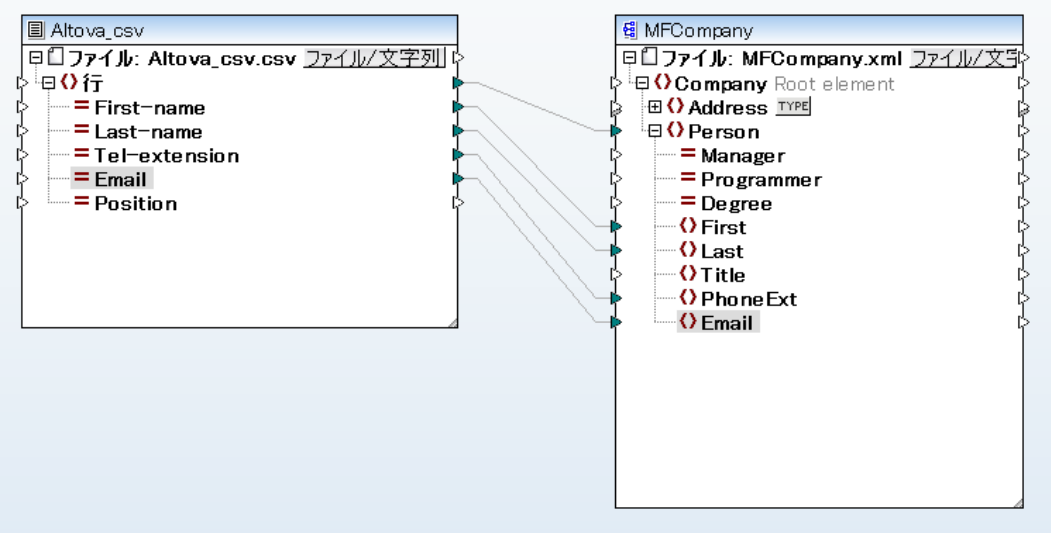
Email、フィールド 5 => Position。ヒント：タブキーを押下することで、フィールド間を移動することができます：ヘッダー1、ヘッダー2 など。



First-name	Last-name	Tel-extension	Email	Position
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Frank	Further	471	f.further@nanonull.com	Accounts Receivable
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager

6. **「OK」**ボタンをクリックします。
7. （プロンプトにて表示される）コンポーネント名を変更ボタンをクリックすることで、コンポーネント名がセットされます。CSV コンポーネントがマッピングに挿入されます。
8. メニューオプションから 挿入 | XML スキーマ/ファイル」を選択し、**MFCompany.xsd** を選択します。
9. **スキップ**をクリックし、ルート要素として Company を選択します
10. 両コンポーネントで対応するアイテムをマッピングします。**行**アイテムをスキーマターゲットの**Person** アイテムへ接続し、出力タブをクリックしてマッピングの結果を確認します。

> CSV コンポーネント内の行アイテムからのスキーマ内の Person アイテムへの接続は、どの要素が反復するか定義するため必要不可欠です。CSV ファイル内の各行に対して、新しい Person 要素が XML 出力ファイル内で作成されることを意味します。



11. 出力タブをクリックして結果を確認します。

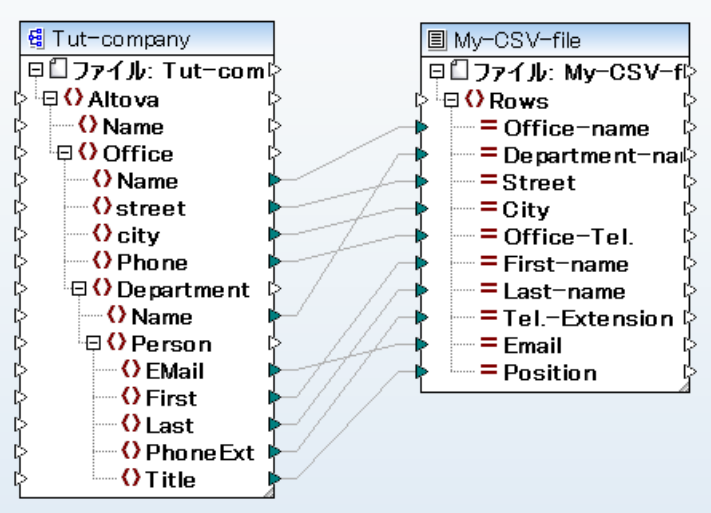


```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3    <Person Manager="true">
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6      <PhoneExt>582</PhoneExt>
7      <Email>v.callaby@nanonull.com</Email>
8    </Person>
9    <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">
```

CSV ファイルからのデータが XML ファイルへマッピングされました。

7.3.2 サンプル :アイテムの反復

ターゲットCSV ファイル内でアイテムの反復 (複数の行)の方法を説明します。このサンプルで使用されるマッピングデザインファイルは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にある **Tut-xml2csv.mfd** で確認できます。

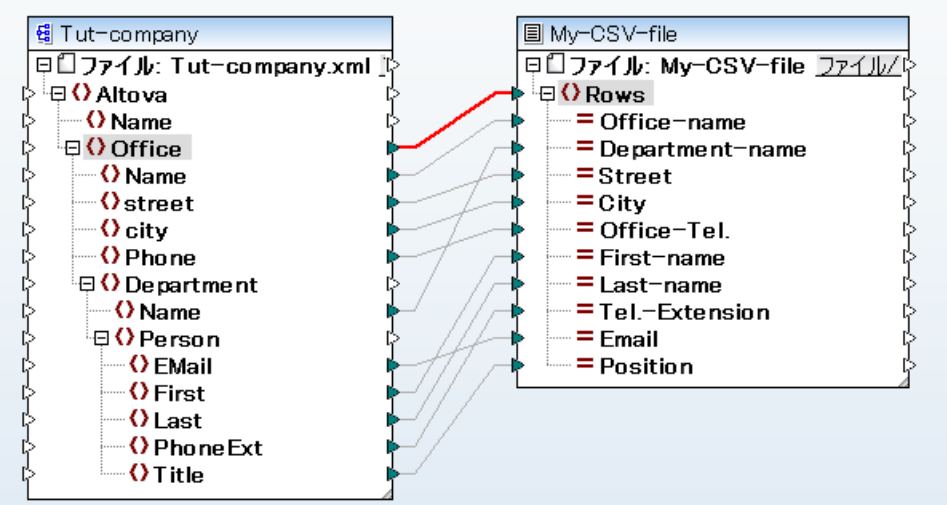


Tut-xml2csv.mfd

このマッピングは、意図的に不完全し作成されています。このサンプルを、メニューコマンド **[ファイル | マッピングの検証]** を使用して検証しようと試みると、検証エラーが生じる可能性があります。また、マッピング出力をプレビューすると、目的に合った、または合わない単一の行が作成されます。

XML ファイル内のアイテムのシーケンスからCSV ファイル内に複数の行を作成することが目的だと仮定します。これは ターゲットCSV ファイルのRows アイテムに接続を描くことで達成することができます。

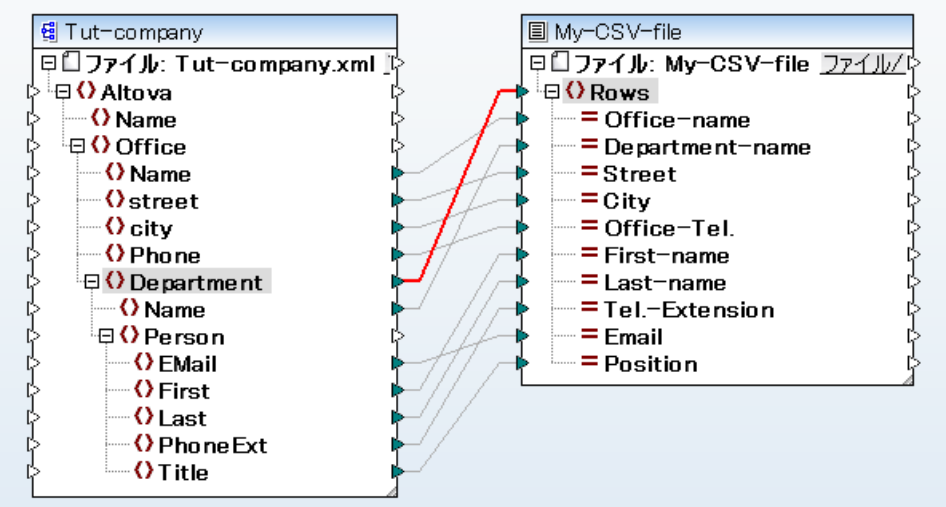
全てのオフィスに関するデータをCSV ファイルへの出力するには、Office を行アイテムへ接続する必要があります。MapForce で以下の命令を与えます :Office アイテムを行アイテムへ接続することで、各 Office (とマッピングされたアイテム) がターゲットCSV ファイルに表示されます。



CSV コンポーネント内のRows アイテムは、接続されたアイテムのシーケンスのための反復子としての役割を果たします。ですから、Office アイテムを接続すると、出力はソースXML 内で見つかる各のために行を作成します。

```
1   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,558833
2   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Otto
3
```

Department から**Rows** アイテムへのマッピングを行うことで、ソース内の各 department のために行が作成されます。



出力は、以下のようになります：

```
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,O
7   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
8
```

最後に、Person をRows アイテムにマッピングすると、結果はすべてのPerson が出力になります。この場合、MapForce は、記録を以下のように反復します：各 Department 内の各 Office 内の各 Person.

7.3.3 サンプル：CSV ならびに固定長テキストファイルから階層構造を作成

このセクションにあるサンプルは**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内部にある**Tut-headerDetail.mfd** に収められています。このサンプルにおけるCSV ファイル(Orders.csv) では、以下にある通り各フィールドに対してそれぞれデータが定義されています：

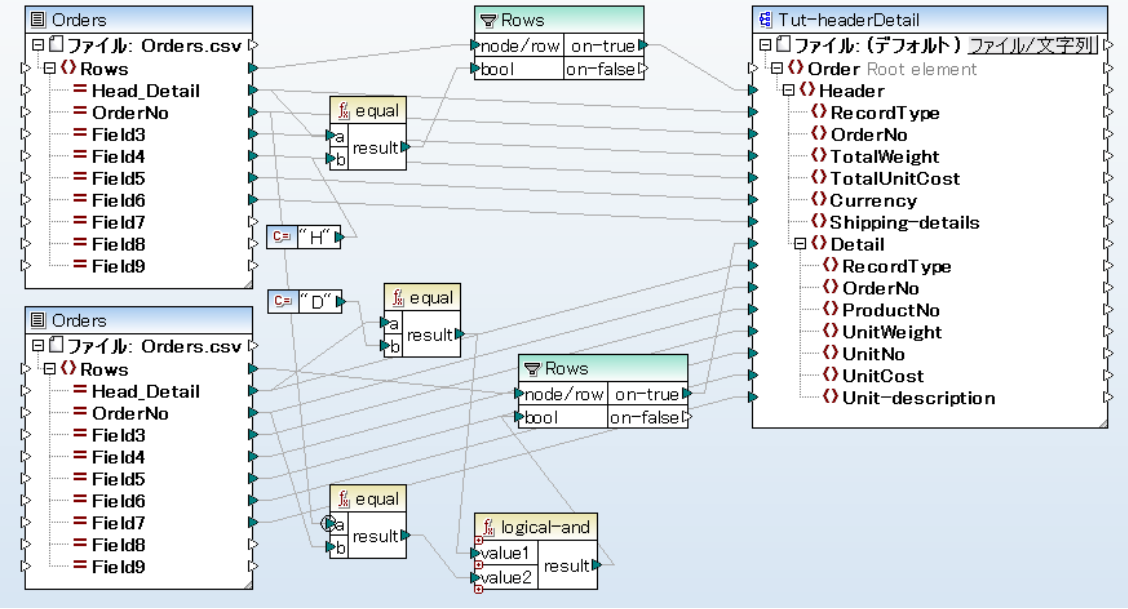
- フィールド 1：H という値はヘッダーが記述されたレコードを表し、D は詳細情報が記述されたレコードを表します。
- フィールド 2：ヘッダーと詳細レコードの共通キーです。
- 各ヘッダー/詳細レコードは別々の行にて表されます。

Orders.csv ファイルの内容を以下に示します。

```
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,,
```

マッピングの目的を以下に示します：

- フラットファイルCSV を階層構造のXML ファイルへマッピングする
- H により表示されるヘッダーレコードをフィルタリングする
- 各ヘッダーレコードに対してD により表される詳細レコードを関連付ける



tut-headerDetail.mfd

上の目的を達成するには、ヘッダーと詳細レコードに共通のフィールドが与えられている必要があります。共通のフィールド（またはキー）とは、この場合 CSV ファイルのレコードであるOrderNo のことになります。CSV ファイルの内容を見ると、最初のヘッダーレコードとそれ以降にある 2 つの詳細レコードに 111 という共通した値が与えられていることが分かります。

Order.csv ファイルが 2 度挿入され、マッピングがより直感的に表示されています。

Tut-headerDetail.xsd スキーマファイルは階層構造により構成されています。Order はルート要素で、子要素に Header を持ち、更にHeader 以下にDetail 要素が配置されています。

最初のOrders.csv コンポーネントからは**Header** レコード（ならびにその他のフィールド）がスキーマターゲットファイルの Header アイテムにマッピングされます。フィルターコンポーネントを使用することで、H レコードのフィルタリングを行います。**行**アイテムにより、フィルタリングされたレコードがスキーマファイル内にあるHeader アイテムに与えられます。

2 番目のOrders.csv コンポーネントからは、Header レコードにあるOrderNo の値にマッチした**Detail** レコードが与えられます。以下の処理が行われます：

- Header レコードにあるOrderNo フィールドと、Detail レコードにあるフィールドがequal 関数により比較されます（パフォーマンスを向上させるため、優先コンテキストはa パラメーターにセットされています）。

- **logical-and** 関数を使用することで、Header レコードの OrderNo と同じ値を含む Detail レコードだけが渡されるようになります。

行アイテムからはフィルターコンポーネントの on-true パラメーターを介して、フィルタリングされたレコードがスキーマコンポーネントの Header ならびに Detail アイテムへ渡されます。

出力タブをクリックすることで、以下に示されるような XML ファイルが表示されます。各ヘッダーレコードには対応するデータが、そして Detail レコードには同じ OrderNo を持つ詳細情報が表示されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
  <Header>
    <RecordType>H</RecordType>
    <OrderNo>111</OrderNo>
    <TotalWeight>332.1</TotalWeight>
    <TotalUnitCost>22537.7</TotalUnitCost>
    <Currency/>
    <Shipping-details>Container ship</Shipping-details>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>A-1579-227</ProductNo>
      <UnitWeight>10</UnitWeight>
      <UnitNo>3</UnitNo>
      <UnitCost>400</UnitCost>
      <Unit-description>Microtome</Unit-description>
    </Detail>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>B-152-427</ProductNo>
      <UnitWeight>7</UnitWeight>
      <UnitNo>6</UnitNo>
      <UnitCost>1200</UnitCost>
      <Unit-description>Miscellaneous</Unit-description>
    </Detail>
  </Header>
```

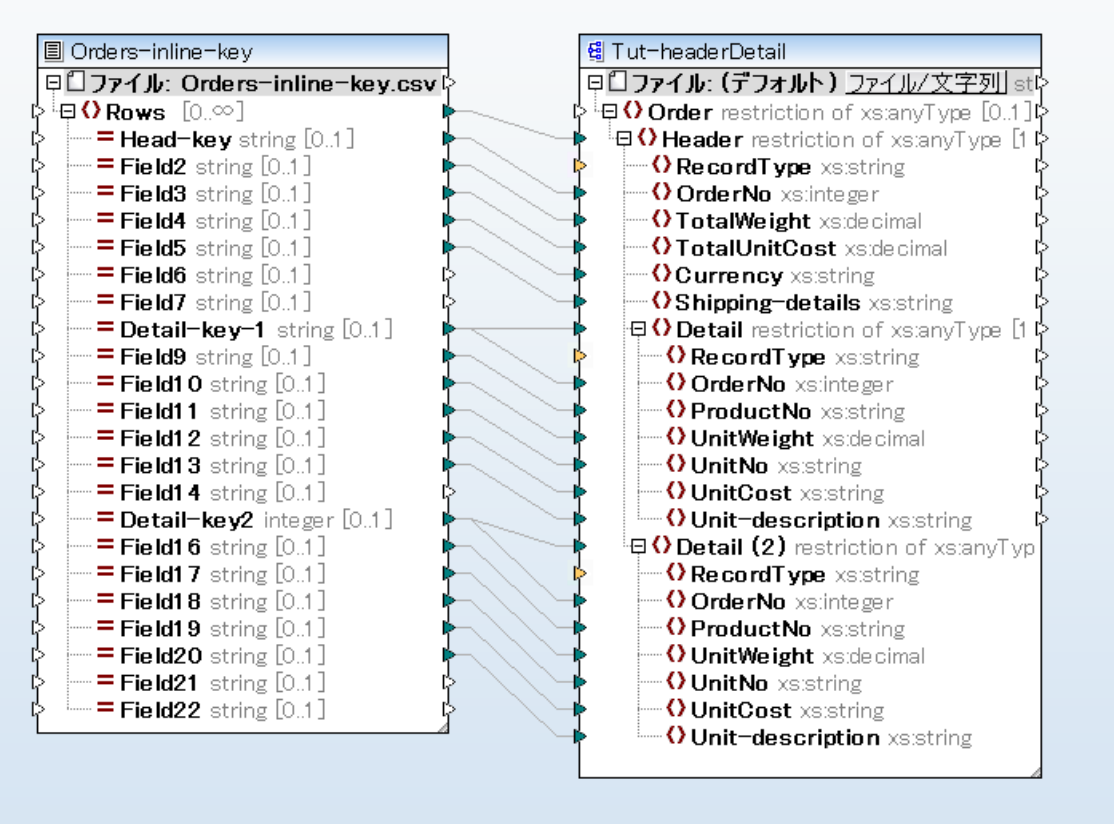
2 番目の例では、別の CSV ファイルを使用します。サンプルは **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーにある **Head-detail-inline.mfd** にて確認します。以下に特徴を示します：

- レコードの識別子（H または D）は利用できません。
- 共通 key フィールドは CSV ファイル内の最初のフィールドに存在しており、ヘッダーならびに詳細レコード（Head-key、Detail-key）が収められています。フィールドはスキーマターゲットの OrderNo へマッピングされます。
- ヘッダーと関連する詳細フィールドは全て同じ行に収められています。

```
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,,,,
```

マッピングは以下のようにデザインされます：

- キーフィールドがスキーマターゲット内の対応する OrderNo アイテムへマッピングされます。
- スキーマターゲットファイル内の Detail アイテムを複製します。この操作により**2 番目**の詳細レコードセットをマッピングすることができるようになります。
- このマッピングの結果として、上にあるサンプルと同様の XML ファイルが作成されます。

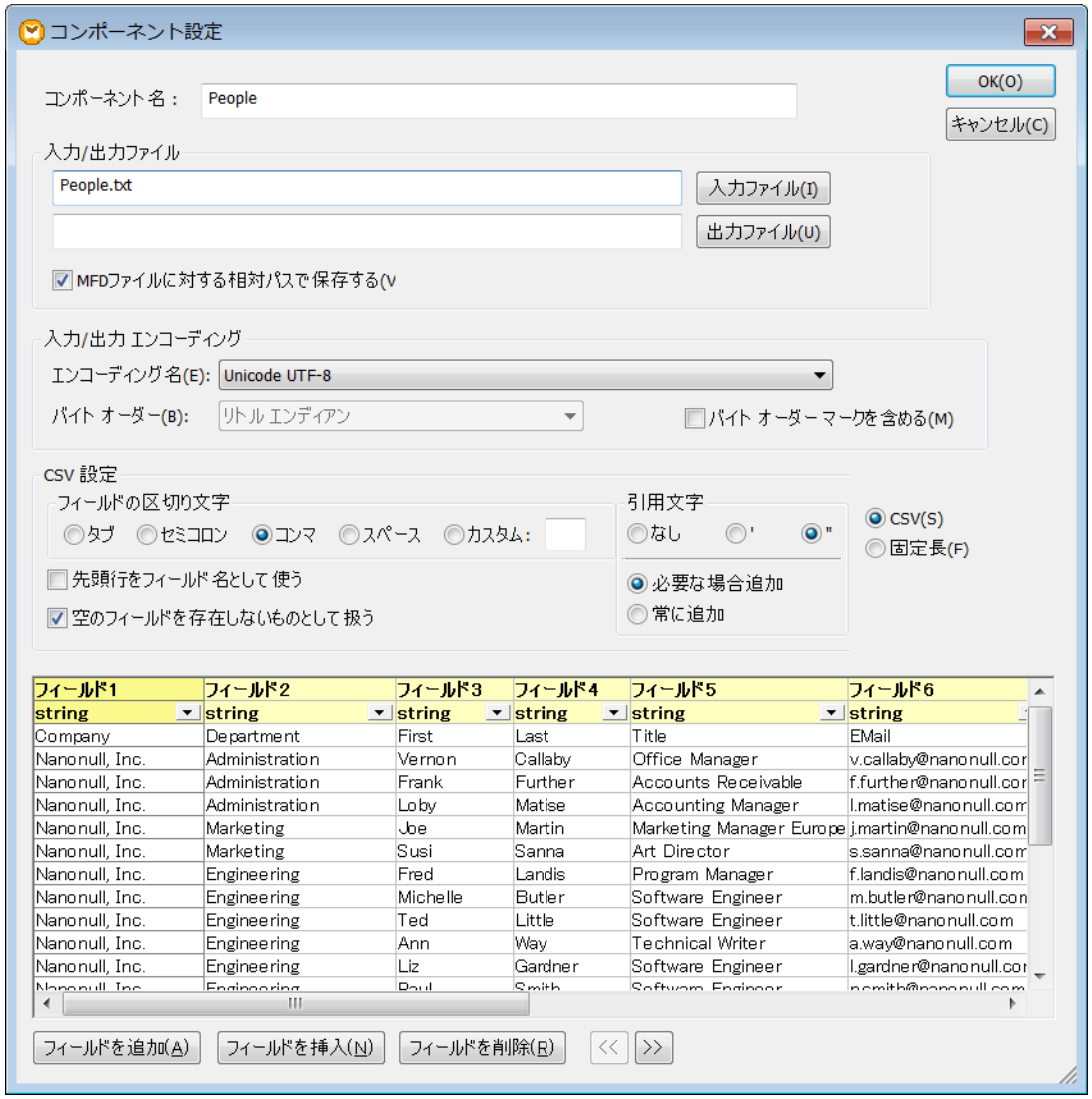


Head-detail-inline.mfd

7.3.4 CSV オプションの設定

テキストコンポーネントをマッピングエリアに追加した後、コンポーネント設定ダイアログボックスで適用できる設定を構成することができます。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます ：

- コンポーネントを選択して、 **「コンポーネント」** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。

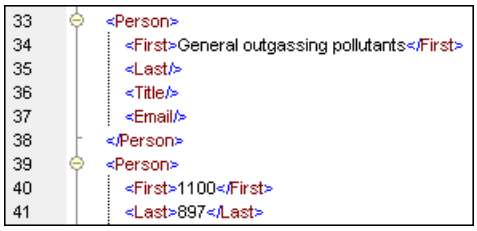


テキストコンポーネント設定ダイアログボックス (CSV モード)

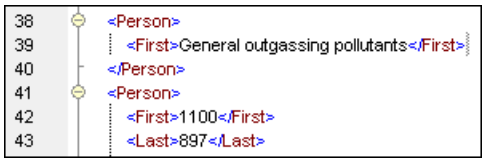
使用することのできる設定は、以下のとおりです。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名が自動的に生成され、インスタンスファイルをその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください： • マッピングをFlowForce Server にデプロイする場合、コンポーネント

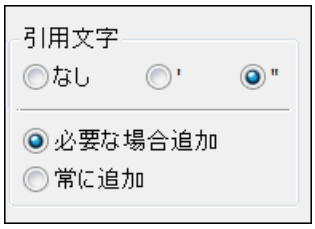
	名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとWindows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
入力ファイル	MapForce がデータを読み込むファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、テキストファイルに割り当てられます。マッピングのためにターゲットとしてテキストファイルが使用されている場合は、フィールドは空のままにしておくことができます。 ソースコンポーネント内で、MapForce は、このフィールドの値を使用し、列名を読み込み、インスタンステキストファイルのコンテンツをプレビューします。 新しいファイルを選択するには、**「入力ファイル」**をクリックします。
出力ファイル	MapForce がデータを書き込むファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 新しいファイルを選択するには、**「出力ファイル」**をクリックします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定は、テキストコンポーネントにより使用された入力と出力ファイルを影響します。以下も参照 : コンポーネント上で相対パスを使用する。
入力 /出力 エンコード	出力インスタンスファイルの次の設定を指定することを許可します : • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。 デフォルトでは、**「新しいコンポーネントのためのデフォルトのエンコード」**オプション内で定義されたエンコードを持つ新しいコンポーネント。**「ツール	オプション」**からこのオプションにアクセスし、タブを生成します。
フィールド区切り記号	テキストファイルの区切りを選択します (CSV ファイルの場合、デフォルトの区切りである ","が使用されます)。このオプションにより、タブ、セミコロン、スペース文字を区切り記号として選択できます。独自の区切り文字をカスタムフィールドにて入力することもできます。
最初の行をフィールドの名前と見なす	テキストファイル内にある最初のレコードの値をカラムのヘッダーとして定義します (プレビューウインドウに表示されます)。テキストコンポーネントがマッピングに表示された際には、カラムヘッダーがアイテム名として表示されます。
空のフィールドを存在しないものとして扱う	このオプションが有効化されると、ソースファイル内の空のフィールドは対応する空のアイテム、(要素または属性) をターゲットファイル内に作成しません。 例 : CSV レコード "`General outgassing pollutants,,,,`" は、4 つのフィールドから構成されており、少なくとも3つのフィールドは空です。 出力が XML ファイルと仮定し、このオプションが無効化されている場合、空の値を持つ空のフィールドが出力に作成されます (このサンプルでは、要素 `Last`、`Title`、と`Email`):

<table>
<tr><td></td><td>


このオプションが有効化されると、空のフィールドは出力に作成されません：



</td></tr>
<tr><td>囲み文字</td><td>入力ファイル内のフィールド値が囲み文字で囲まれている場合、ソースファイルと同じ囲み文字を選択してください。同じ設定が出力ファイルにも使用されます。



出力ファイルのために、追加設定を指定することができます：

必要な場合追加する フィールド区切り記号、または改行を含むテキストのフィールドのみに選択された囲み文字を追加します。

常に追加する 生成されたCSV ファイルのすべてのフィールドに選択された囲み文字を追加します。</td></tr>
<tr><td>CSV / 固定値</td><td>コンポーネントの型を CSV または FLF (固定値フィールド)に変更します。</td></tr>
<tr><td>プレビューエリア</td><td>ダイアログボックスの下の部分では、入力または出力として選択されたファイルの 20 の行を表示します。

必要であれば、以下のように、ファイルの構造を作成 (または既存の構造を変更) することができます。

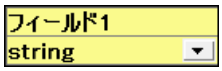
フィールドの追加 最後のCSV レコードの後に、新規のフィールドを作成します。

フィールドの挿入 現在選択されているCSV レコードの前に、新規のフィールドをすぐに作成します。

フィールドの削除 現在選択されているフィールドを削除します。

<< 現在選択されているフィールド左に1つ移動します。

>> 現在選択されているフィールド右に1つ移動します。</td></tr>
</table>

	フィールドの名前を変更するには、ヘッダーをクリックして (例 : **フィールド1**)、新しい値を入力します。**最初の行をフィールドの名前とみなす** オプションが有効化されている場合、フィールド名は編集することができません。  ドロップダウンリストから必要とされる値を選択し、フィールドのデータ型を変更します。MapForce は、入力データとフィールドフォーマットが一致するかデータ型をチェックします。フィールド フォーマットが一致しない場合、はデータが赤では以来とされます。  フィールド型はデフォルトの XML スキーマデータ型をベースにしています。例 : データ型は以下のフォームで表示されます :**YYYY-MM-DD**。

7.3.5 サンプル :固定長テキストファイルを (データベースへ) マッピング

このサンプルでは固定長テキストファイル (FLF) とMS Access データベースの間のマッピングを行います。このサンプルで使用されるファイルは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーに収められています。ソーステキストファイルとターゲット データベースは、従業員のリストを保管しています。ソースファイル内では、記録は明示的にサイズにより以下のようにカンマで区切られています :

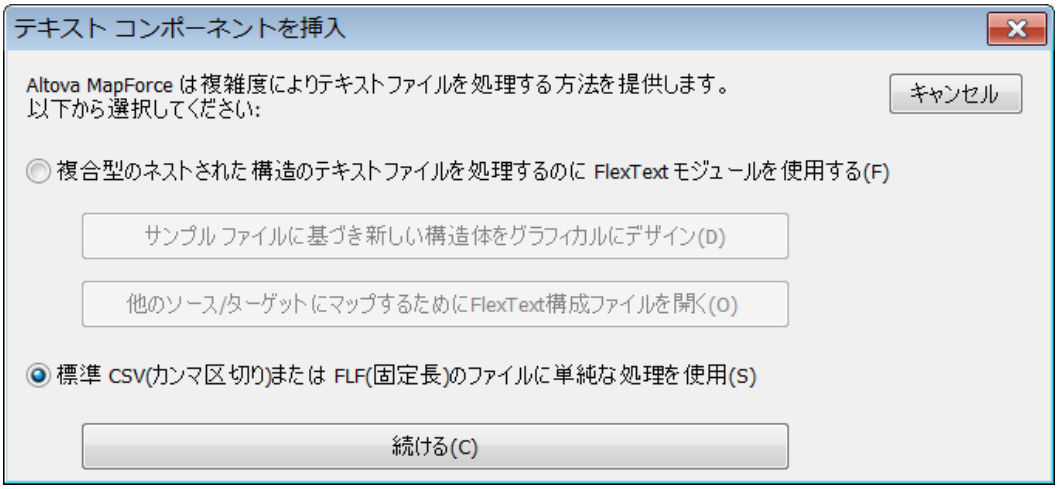
フィールドの位置と名前	サイズ (文字数)
Field 1 (First name)	8
Field 2 (Last name)	10
Field 3 (Phone extension)	3
Field 4 (Email)	25
Field 5 (Position)	25

マッピングの目的は、データベース内の各社員の内線番号を、各内線番号のプレフィックスに "100" を追加し、ソースファイル内のものに更新することです。このゴールを達成するには、以下を行います :

ステップ 1 : テキストコンポーネントの挿入と構成
ステップ 2 : データベースコンポーネントを挿入する
ステップ 3 : マッピングのデザイン
ステップ 4 : マッピングの実行

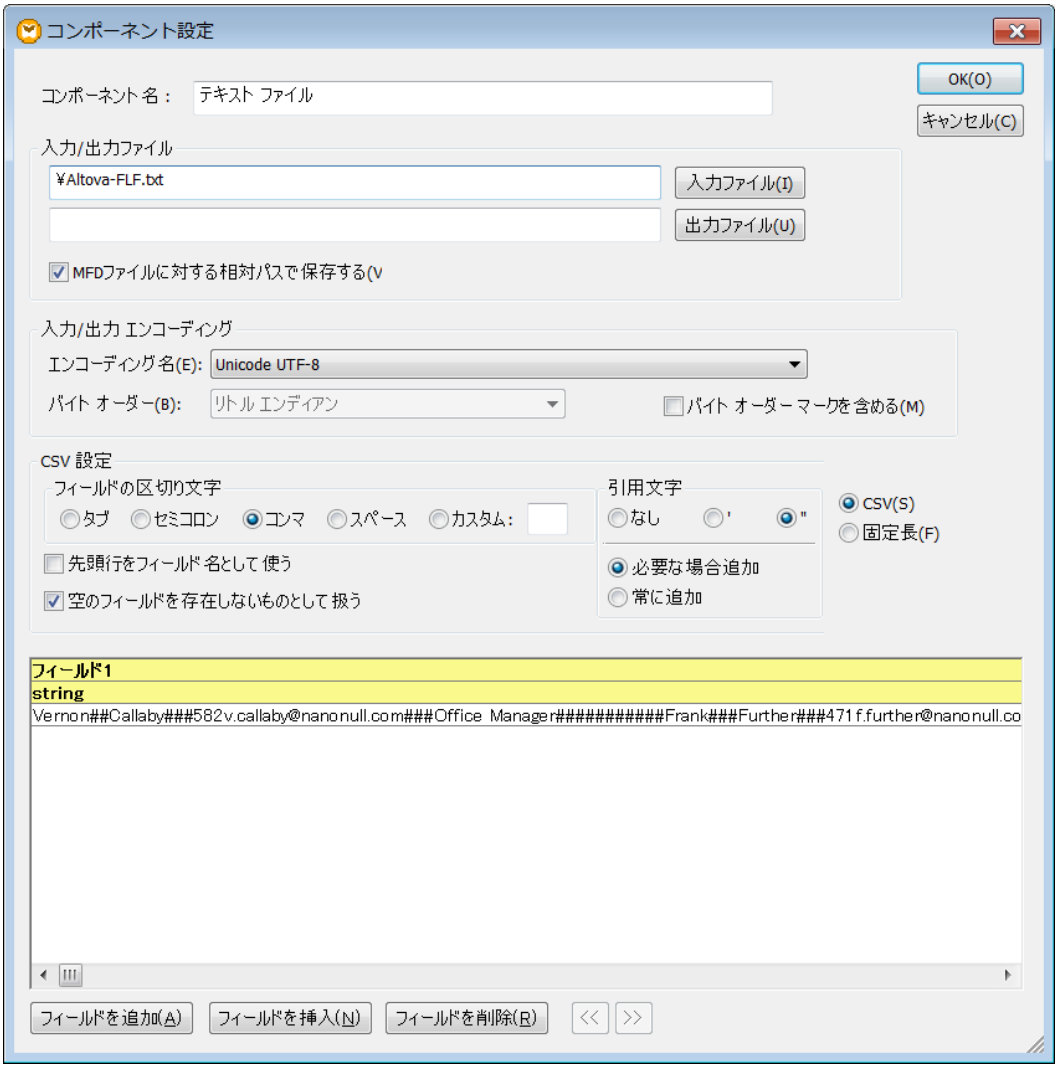
ステップ 1 : テキストコンポーネントの挿入と構成

1. メニューオプションの 挿入 | テキストファイル を選択するか、テキストファイルを挿入アイコン をクリックします。テキストコンポーネントを挿入ダイアログボックスが表示されます。

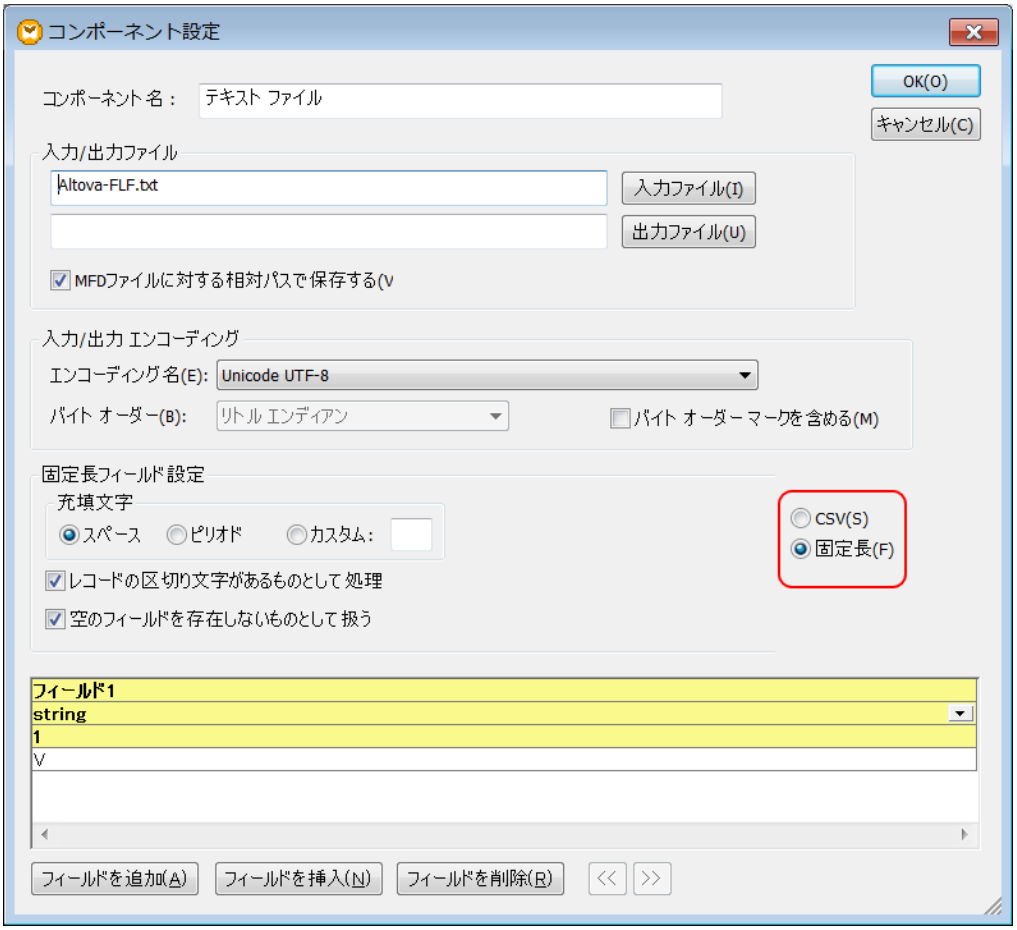


単純な処理を使用 ...を選択して、「続ける」ボタンをクリックします。

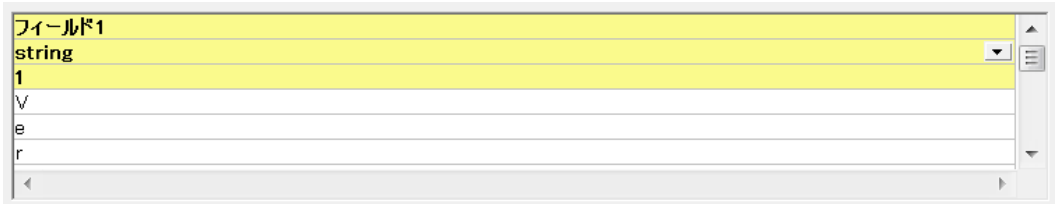
2. 入力ファイルボタンをクリックして **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Altova-FLF.txt** ファイルを選択します。ファイルが 1 つの 大きな 文字列と # の充填文字から構成されていることが確認できます。


コンポーネント設定
コンポーネント名：
テキスト ファイル
OK(O)
キャンセル(C)
入力/出力ファイル
¥Altova-FLF.txt
入力ファイル(I)
出力ファイル(U)
MFDファイルに対する相対パスで保存する(V
入力/出力 エンコーディング
エンコーディング名(E):
Unicode UTF-8
バイト オーダー(B):
リトル エンディアン
バイト オーダー マークを含める(M)
CSV 設定
フィールドの区切り文字
タブ
セミコロン
コンマ
スペース
カスタム:
引用文字
なし
'
"
CSV(S)
固定長(F)
先頭行をフィールド名として使う
空のフィールドを存在しないものとして扱う
必要な場合追加
常に追加
フィールド1
string
Vernon##Callaby###582v.callaby@nanonull.com###Office Manager##########Frank###Further###471f.further@nanonull.co
フィールドを追加(A)
フィールドを挿入(N)
フィールドを削除(R)

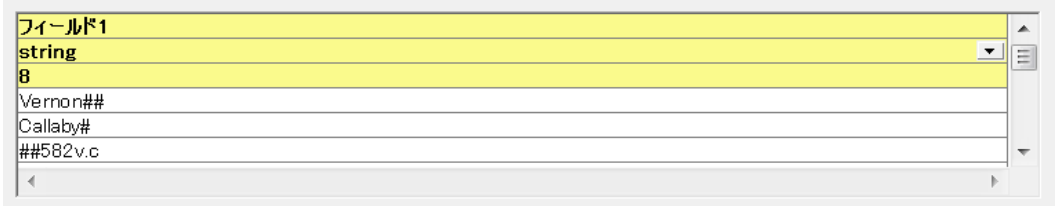
3. **固定長**を選択します。



4. **レコードの区切り文字があるものとして処理**チェックボックスのチェックを外します。



5. 黄色でハイライトされた3行は編集することができます。また、以下を指定することができます :i) フィールド名、ii) データ型、iii) フィールドのサイズ。値を 8 に変更した後 Enter キーを押下します。 8文字ごとにフィールドが区切られるため、より多くのデータが最初のレコードに表示されます。



6. **フィールドを追加**ボタンをクリックして、新たなフィールドを追加します。2番目のフィールド長さを10文字に指定します。

フィールド1	フィールド2
string	string
8	10
Vernon##	Callaby###
582v.cal	laby@nanon
ull.com#	##Office M

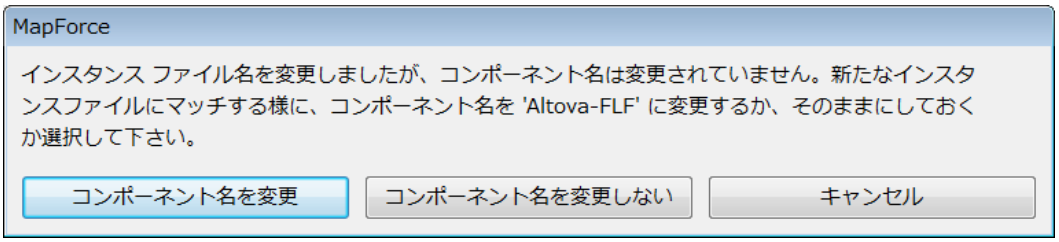
7. 同様の方法を使用して、更に3つのフィールドを作成します。長さはそれぞれ 3、25、25 文字になります。更にフィールドの名前をそれぞれ First、Last、Tel.-Ext、Email、Title とすることで、簡単にマッピングを行えるようにします。プレビューは以下のように表示されます：

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
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager
Joe	Firstbread	621	j.firstbread@nanonull.com	Marketing Manager Europ
Susi	Sanna	753	s.sanna@nanonull.com	Art Director
Fred	Landis	951	f.landis@nanonull.com	Program Manager

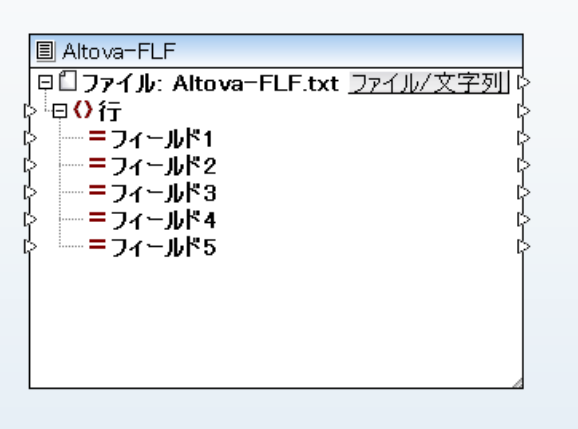
8. 固定長フィールド設定グループのカスタムテキストボックスをクリックして、ハッシュ文字 (#) を入力します。この操作により、充填文字として使用されている文字が入力文字列から取り除かれます。



9. 「OK」をクリックします。

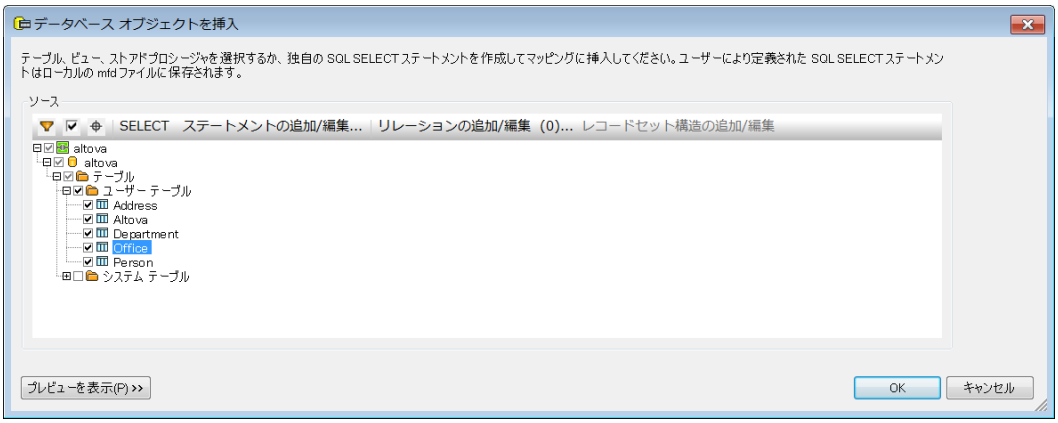


10. 「コンポーネント名を変更」ボタンをクリックして、名前を変更します。テキストファイルコンポーネントがマッピングウィンドウに表示されます。このコンポーネントを使ったデータのマッピングが行えるようになります。



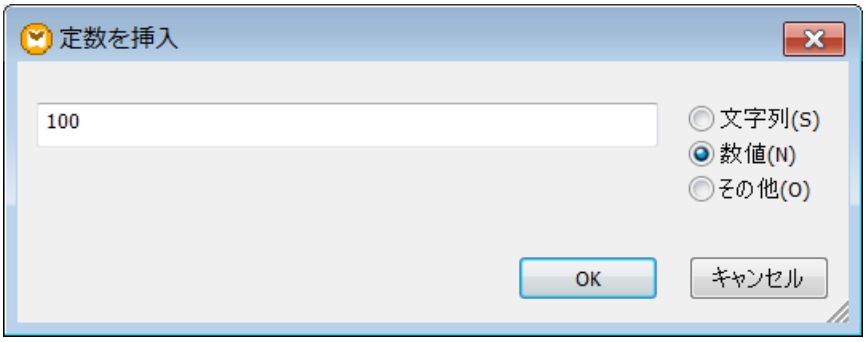
ステップ 2：データベースコンポーネントの挿入

1. メニューオプションの 挿入 | データベース」を選択し、**Microsoft Access** ラジオボタンを選択して **「次へ」** をクリックします。
2. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内にある **altova.mdb** データベースを選択し、 **「接続」** をクリックします。
3. Person テーブルのチェックボックスを選択し、 **「OK」** をクリックします。

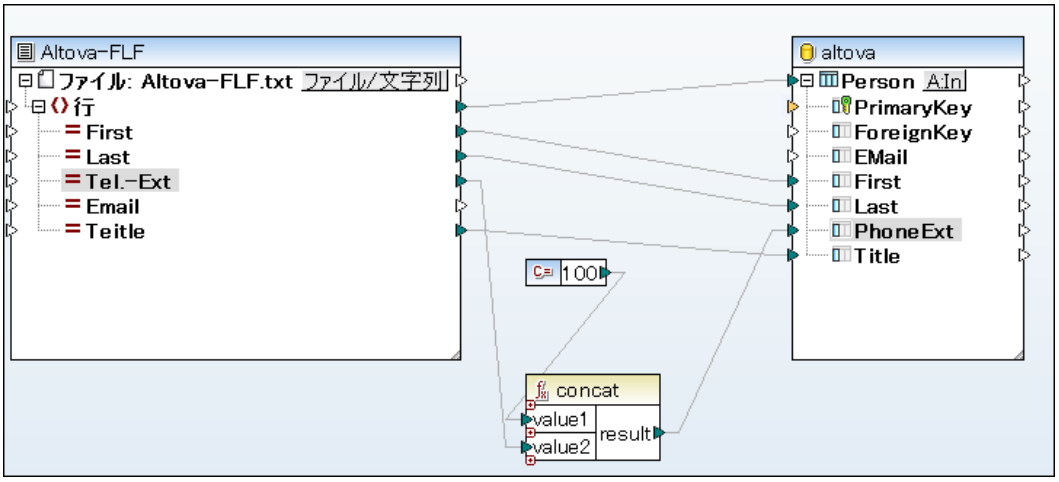


ステップ 3：マッピングのデザイン

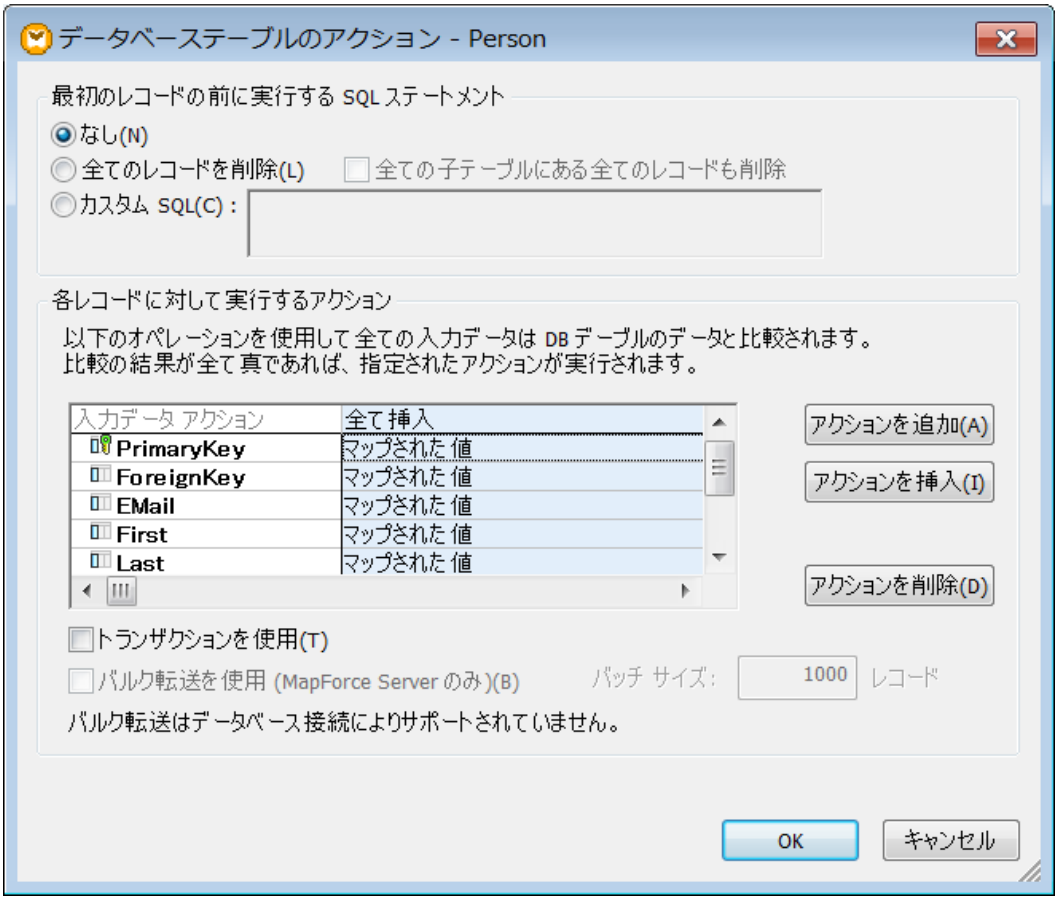
1. `core | concat` 関数をライブラリウィンドウからマッピングタブへドラッグします。
2. メニューオプションの 挿入 | 定数」を選択し、数値ラジオボタンをクリックした後に、新たな内線番号のプレフィックスとして使用する 100 を入力します。この定数が内線番号のプレフィックスを保管します。



3. 以下のスクリーンショットに示されるマッピングを作成します。



4. **Person** エントリーの隣にあるテーブルアクションボタン A:In をクリックします。
5. **「入力データアクション」**のコンボボックスをクリックし、**「以下の条件で更新」**エントリーを選択し、**First** と **Last** のフィールドが**一致**するように設定します。この操作により、**First** の内容がソーステキストファイルとデータベースにて一致する場合のみ、Person テーブルが更新されるようになります。条件が真の場合に取られるアクションはマッピングにより定義されており、この場合は "100" というプレフィックスが追加された内線番号が**Person** テーブルの**PhoneExt** フィールドに与えられます。



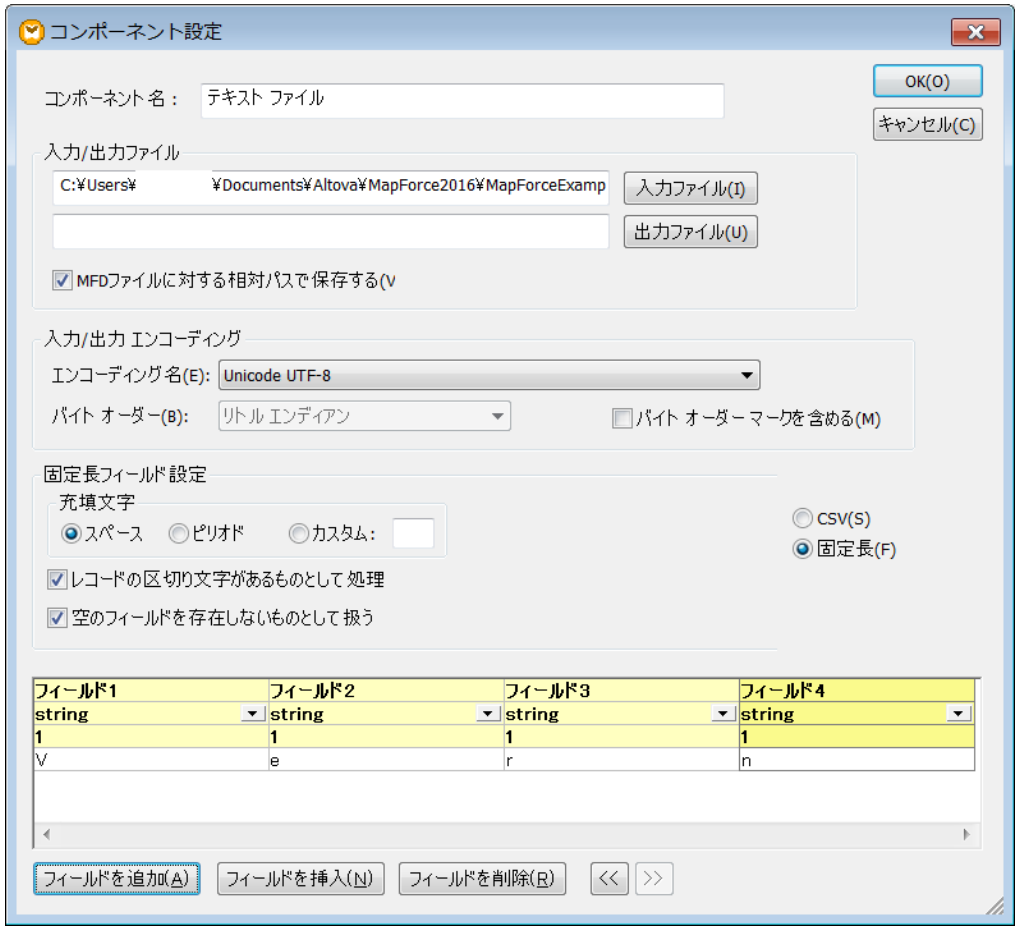
ステップ 4 : マッピングの実行

出力タブをクリックして、生成されたSQL ステートメントをプレビューします。SQL スクリプトの実行ボタン をクリックして、SQL ステートメントを実行します。

7.3.6 FLF オプションの設定

テキストコンポーネントをマッピングエリアに追加した後、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます :

- コンポーネントを選択して、 **「コンポーネント」** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。



テキストコンポーネント設定ダイアログボックス (固定値フィールド モード)

使用することのできる設定は、以下のとおりです .

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名が自動的に生成され、インスタンスファイルをその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符を含んではなりません。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください: • マッピングをFlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとWindows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
入力ファイル	MapForce がデータを読み込むファイルを指定します。このフィールドはソースコ

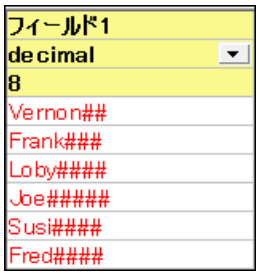
	ンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、またテキストファイルに割り当てられます。テキストファイルコンポーネントをマッピングのためにターゲットとして、使用する場合、フィールドは空にしてください。 ソースコンポーネント内で、MapForce はこのフィールドの値を使用して、列名を読み込み、インスタンステキストファイルのコンテンツをプレビューします。 新しいファイルを選択するには、**「入力ファイル」**をクリックします。
出力ファイル	MapForce がデータを書き込むファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 新しいファイルを選択するには、**「出力ファイル」**をクリックします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定は、テキストコンポーネントにより使用された入力と出力ファイルに影響を与えます。以下も参照：コンポーネント上で相対パスを使用する。
入力 /出力 エンコード	出力インスタンスファイルの次の設定を指定することを許可します： • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。 デフォルトでは、**「新しいコンポーネントのためのデフォルトのエンコード」**オプション内で定義されたエンコードを持つ新しいコンポーネント。**「ツール	オプション」**からこのオプションにアクセスし、タブを生成します。
充填文字	データの長さがフィールドで定義されたものよりも短い場合、このオプションで指定された文字により残りのスペースが充填されます。カスタムフィールドにより独自の充填文字を定義することができます。 カスタムフィールドに入力された充填文字と同じ文字が入力データに含まれている場合、入力データにある文字も充填文字として切り取られます！
レコードの区切り文字があるものとして処理	このオプションは、ソースフラットファイルからCR/LFなどの区切り記号 を含まないデータを読み込むために役に立ちます。または、区切り文字を含まないターゲットフラットFLF ファイルを作成する際にも役に立ちます。 下の レコードの区切り文字があるものとして処理するオプション セクションを参照してください。
空のフィールドは存在しないものとして扱う	このオプションが有効化されると、ソースファイル内の空のフィールドは対応する空のアイテム (要素または属性) をターゲットファイル内に作成しません。 出力がXML ファイルと仮定し、このオプションが無効化されている場合、空の値を持つ空のフィールドが出力に作成されます (このサンプルでは、要素 `Last`、`Title`、と`Email`):

<table>
<tr><td></td><td>
<pre>
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>
</pre>
このオプションが有効化されると、空のフィールドは出力に作成されません：
<pre>
38 <Person>
39   <First>General outgassing pollutants</First>
40 </Person>
41 <Person>
42   <First>1100</First>
43   <Last>897</Last>
</pre>
</td></tr>
<tr><td>CSV / 固定</td><td>コンポーネントの型を CSV または FLF (固定値フィールド)に変更します。</td></tr>
<tr><td>プレビューエリア</td><td>ダイアログボックスの下の部分では、入力または出力として選択されたファイルの 20 の行を表示します。

必要であれば、以下のように、ファイルの構造を作成 (または既存の構造を変更) することができます。

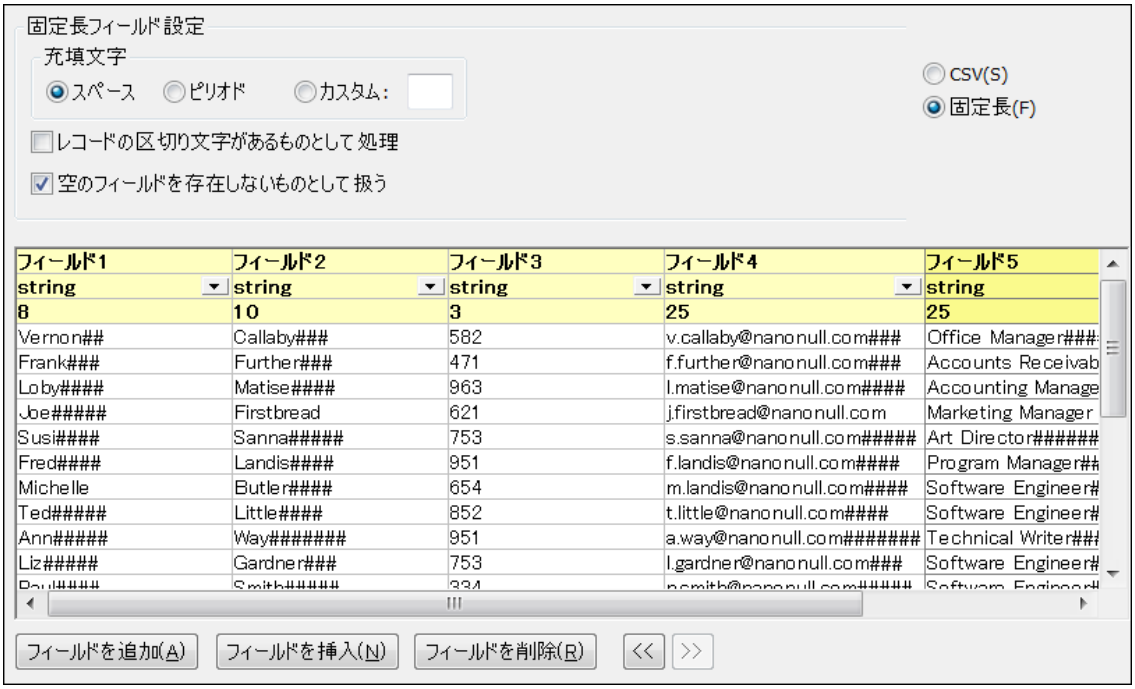
フィールドの追加 最後のレコードの後に新規のフィールドを作成します。
フィールドの挿入 現在選択されている記録の前に、新規のフィールドをすぐに作成します。
フィールドの削除 現在選択されているフィールドを削除します。
<< 現在選択されているフィールド左に1つ移動します。
>> 現在選択されているフィールド右に1つ移動します。

フィールドの名前を変更するには、ヘッダーをクリックして (このサンプルでは、フィールド1)、新しい値を入力します。
<table>
<tr><td>フィールド1</td></tr>
<tr><td>string</td></tr>
<tr><td>8</td></tr>
<tr><td>Vernon##</td></tr>
<tr><td>Frank###</td></tr>
<tr><td>Loby####</td></tr>
<tr><td>Joe#####</td></tr>
<tr><td>Susi####</td></tr>
<tr><td>Fred####</td></tr>
</table>
フィールドのデータ型を変更するには、ドロップダウンリストから必要な値を選択してください。MapForce は、入力データとフィールドフォーマットが一致しない場合赤でハイライトできるように、データ型をチェックします。</td></tr>
</table>

	 フィールド内の文字のサイズを設定できるように、上から3番目にフィールドのサイズを入力します。

"レコードの区切り文字があるものとして処理" オプションを理解する

このオプションを理解するためには、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内の **Altova-FLF.txt** ファイルを開きます。ファイルは、CR/LF などの区切り文字を使用しない、71文字の長さのレコードから構成されています。このファイルからデータを読み込む場合、最初にこのファイルをレコードに分割する必要があります。 すなわち、下に表示されるようにフィールドが71文字である複数のフィールドを作成し、**レコードの区切り文字があるものとして処理** を無効化します。順序を追った例は、 例 :固定長のテキストファイルをデータベースにマッピングするを参照してください。



このファイルから同じ構造を使用する目的のファイルにデータを書き込む場合、**"レコードの区切り文字があるものとして処理"**を有効化すると、新規のレコードが72文字目から新たに表示されます。

```
 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#
 5   Susi####Sanna#####753s.sanna@nanonull.com#####Art Director##############
 6   Fred####Landis####951f.landis@nanonull.com####Program Manager###########
 7   MichelleButler####654m.landis@nanonull.com####Software Engineer########
 8   Ted#####Little####852t.little@nanonull.com####Software Engineer########
 9   Ann#####Way#######951a.way@nanonull.com#######Technical Writer#########
10   Liz#####Gardner###753l.gardner@nanonull.com###Software Engineer########
11   Paul####Smith#####334p.smith@nanonull.com#####Software Engineer########
12   Alex####Martin####778a.martin@nanonull.com####IT Manager###############
13   George##Hammer####223g.hammer@nanonull.com####Web Developer############
14   Jessica#Bander####241j.band@nanonull.com######Support Engineer#########
15   Lui#####King######345l.king@nanonull.com######Support Engineer#########
16   Steve###Meier#####114s.meier@nanonull.com#####Office Manager###########
17   Theo####Bone######331t.bone@nanonull.com######Accounts Receivable######
18   Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager US
19   ValentinBass######716v.bass@nanonull.com######IT Manager###############
20   Carl####Franken###147c.franken@nanonull.com###Support Engineer#########
21   Mark####Redgreen##152m.redgreen@nanonull.com##Support Engineer#########
22
```

"レコードの区切り文字があるものとして処理"が有効化された場合のマッピングの結果

"レコードの区切り文字があるものとして処理"が無効化されている場合、マッピングされたデータが、充填文字とともに、1 つの長い文字列として表示されます。

```
1    Vernon##Callaby###582v.callaby@nanonull.com###Office
     Manager###########Frank###Further###471f.further@nanonull.com###Accounts
     Receivable######Loby####Matise####963l.matise@nanonull.com####Accounting
     Manager#######Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager
     Europe#Susi####Sanna#####753s.sanna@nanonull.com#####Art
     Director#############Fred####Landis####951f.landis@nanonull.com####Program
     Manager##########MichelleButler####654m.landis@nanonull.com####Software
     Engineer########Ted#####Little####852t.little@nanonull.com#####Software
     Engineer########Ann#####Way#######951a.way@nanonull.com######Technical
     Writer#########Liz#####Gardner###753l.gardner@nanonull.com###Software
     Engineer########Paul####Smith#####334p.smith@nanonull.com#####Software
     Engineer########Alex####Martin####778a.martin@nanonull.com####IT
     Manager###############George##Hammer####223g.hammer@nanonull.com####Web
     Developer############Jessica#Bander####241j.band@nanonull.com######Support
     Engineer#########Lui#####King######345l.king@nanonull.com######Support
     Engineer#########Steve###Meier#####114s.meier@nanonull.com#####Office
     Manager###########Theo####Bone######331t.bone@nanonull.com#####Accounts
     Receivable######Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager
     USValentinBass######716v.bass@nanonull.com######IT
     Manager###############Carl####Franken###147c.franken@nanonull.com###Support
     Engineer#########Mark####Redgreen##152m.redgreen@nanonull.com##Support
     Engineer#########
```

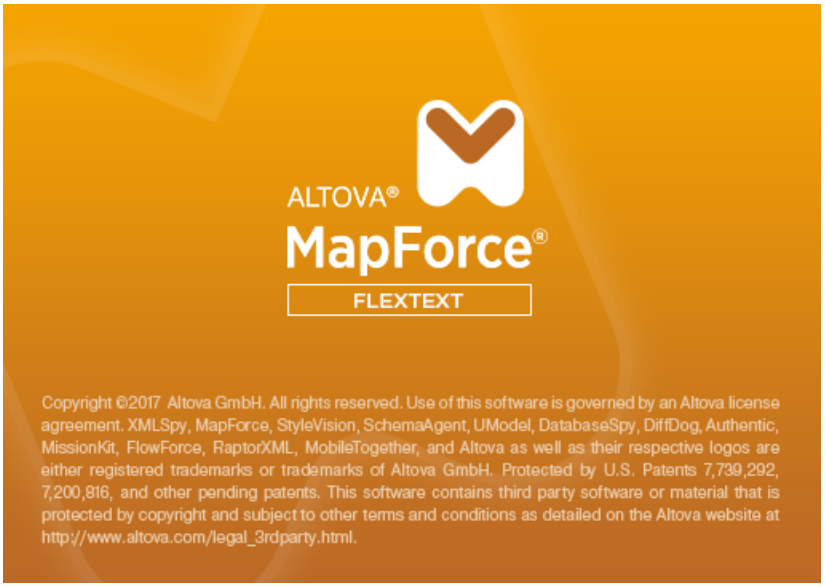
"レコードの区切り文字があるものとして処理" が無効化された場合のマッピング結果

7.4 MapForce FlexText

Altova Web サイト 複雑な テキストファイルのマッピング -FlexText

FlexText は、複雑な非標準、または、レガシーテキストファイルからデータを MapForce によりサポートされる他のフォーマットに変換する MapForce モジュールです。XML ファイルには、 MapForce がスキーマから抽出することのできる スキーマ、または、構造が存在しますが、テキストファイルは異なります。特に、CSV または 固定 長さ フィールド パターンに当てはまらない複雑かつユニークな構造を持つ場合があります。更に、レガシーテキストファイルから必要な一部のデータを抽出する必要がある場合があります。FlexText はこれらの問題をルールまたはテンプレートを視覚的に定義し、テキストデータをマップすることのできるアイテムに分割して解決します。FlexText テンプレートは、基本的には、他のフォーマットにマッピングする目的のため、指定する条件に従いカスタムテキストデータの構造モデルを定義します。

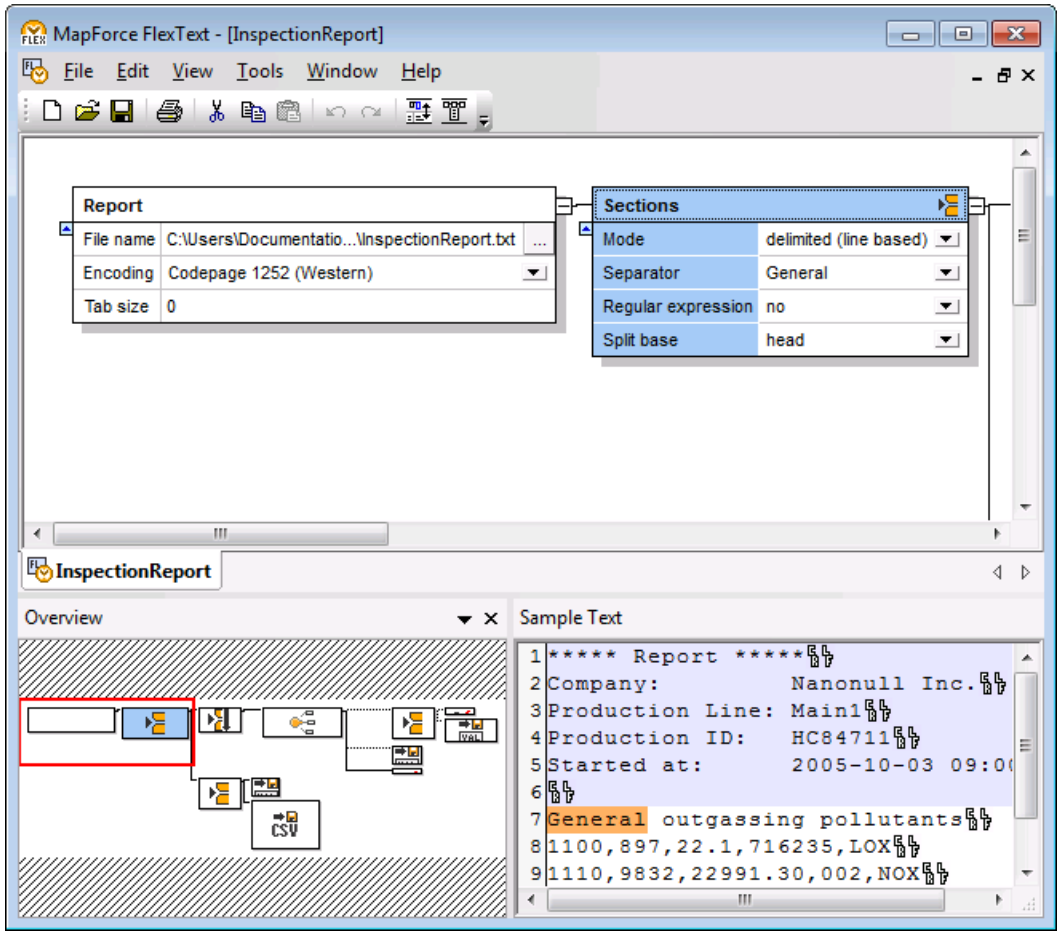
FlexText テンプレートを1度定義すると、MapForce マッピングエリアにソースまたはターゲットコンポーネントとして追加し、MapForce によりサポートされる他のフォーマットに複雑で標準では内テキストデータをマップすることができます。FlexText テンプレート ファイルには、 .mft (MapForce FlexText) 拡張子があります。複数のマッピ内で同じFlexText テンプレートを再利用し、多数のテキストファイルを処理することができます。



7.4.1 概要

(**挿入 | テキストファイル**) メニューコマンドを使用して)マッピング コンポーネントとしてテキストファイルを追加する際に、MapForce から、 FlexText を直接開始することもできます。MapForce インストールディレクトリ内の *Altova MapForce FlexText* 実行可能ファイルとして実行することにより、FlexText をスタンドアロンプログラムとして開始することができます。

下のスクリーンショットは、 **<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリ内にある他の FlexText サンプルと共にサンプル FlexText テンプレート (**InspectionReport.mft**) を表しています。



FlexText グラフィカル インターフェイス

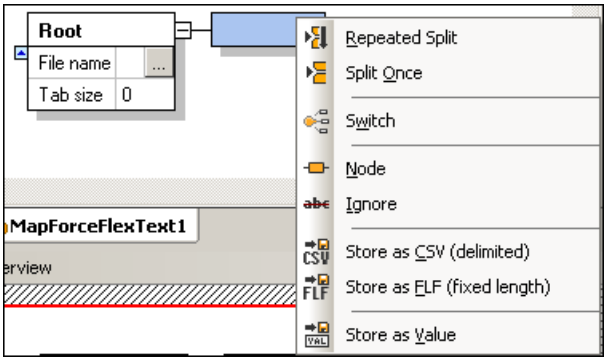
FlexText インターフェイスは次のペインにより構成されています：デザイン ペイン、概要ペイン、とサンプルテキストペイン。次に示されるように、これらのペインの位置と外観は、優先順位によりカスタム化することができます。

デザイン ペイン

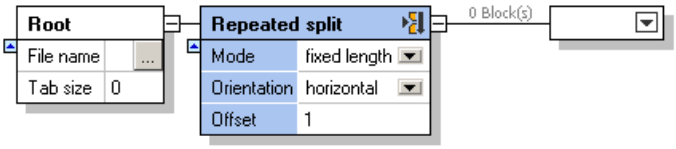
デザイン ペインは、コンテナを使用して、テキストの構造を定義する作業エリアです。 コンテナは、指定する条件に従い、テキストデータを扱うためのルールを定義します。FlexText テンプレートの最初のコンテナは、全体のテキストファイルを示す *ルート* コンテナです。すべての他の コンテナ は、 (通常、テキストを意味のある単位に分割して、データをマップするため) 次のコンテキストの処理ロジックを定義します。例えば、 *「1回分割する」* コンテナは、テキストのフラグメントを 2つのフラグメントに分割します。デフォルトでは、コンテナの名前は、関数を説明しています (例えば、*繰り返される分割*)。しかしながら、必要に応じて名前を変更することができます。作業中のテキストファイルの構造により必要とされる数のコンテナを定義することができます。

このドキュメントの FlexText リファレンス セクションは、コンテナについての詳細をカバーしています。しかしながら、コンテナに関しては、以下に注意してください：

- コンテナの右上の角のアイコンをクリックすると、ポップアップメニューが開かれ、コンテナ型を選択することができます。



- 各コンテナには、構造内の特定の箇所でテキストデータに何が行われるかを指定することができます。このオプションにより、コンテナのコンテンツを決定し、MapForce コンポーネントに与える前に設定を直すことができます。



- コンテナコンテンツを表示、または、非表示するには、 **「展開」** または **「折りたたむ」** アイコンをそれぞれクリックします。
- グループとしてコンテナを折りたたむには、**Shift** キーを押してください。2つの二重山かっこ が表示されます。ハンドルをクリックすると、コンテナツリーのセクションクリックされたアイテムの右側に折りたたみます。
- デザインペイン内で、特定のコンテナにより作成されたテキストをプレビューするには、 **「デザインビュー内のノードテキスト」** ツールバーボタンを有効化してください。
- **「選択されていないノードテキストを自動的に折りたたむ」** ツールバーボタンが有効化されていると、コンテンツof a コンテナ のコンテンツは、選択されると、展開されます。すべての他の コンテナハ、折りたたまれます。

概要ペイン

概要ペインは、デザインペインの概観図を表示しています。デザインペイン内を移動するには、赤い三角をクリックアンドドラッグします。概要ペインをデタッチして、インターフェイス内で位置を変更するには、タイトルバーをクリックして、ペインを希望する場所にドラッグします。

サンプルテキストペイン

サンプルテキストペインは、現在選択されている コンテナのコンテンツを表示します。 (このペイン **「デザインビュー内のノードテキスト」** ツールバーボタンが有効されていると、表示されず、代わりに、現在選択されている コンテナ のコンテンツがデザインペインに表示されます)。

テキスト内のプレビューで、 **「タブ」** と **「Enter キー改行」** 文字を素早く識別するために、FlexText は、このような文字をそれぞれ と などの記号を使用して表示します。

7.4.2 FlexText チュートリアル

このチュートリアルは、FlexText の最も一般的な使用方法と、MapForce 内で多種の方法でテキストファイルを処理するための FlexText の最もパワフルな機能をについて説明します。

このサンプルは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内の **Flex-tutorial.txt** ファイルを使用します。 txt ファイルは次の書式で表示されます :

```
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
```

チュートリアルの目的 :

- 111 と 222 のキーを含むレコードを個別にマップ可能なアイテムに分離する。
- テキスト形式のレコードを破棄する。
- 残りのレコードの CSV ファイルを作成する。

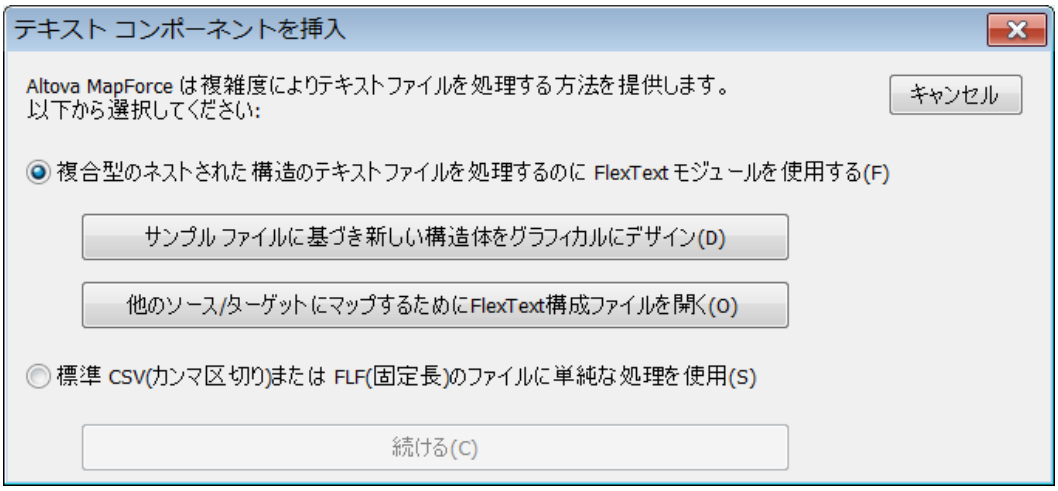
チュートリアルは、次のパートから構成されています :

- ステップ 1 :FlexText テンプレートの作成
- ステップ 2 :分割の条件の定義
- ステップ 3 :各コンテナのための複数の条件の定義
- ステップ 4 :ターゲット MapForce コンポーネントの作成
- ステップ 5 :MapForce 内で FlexText テンプレートを使用する

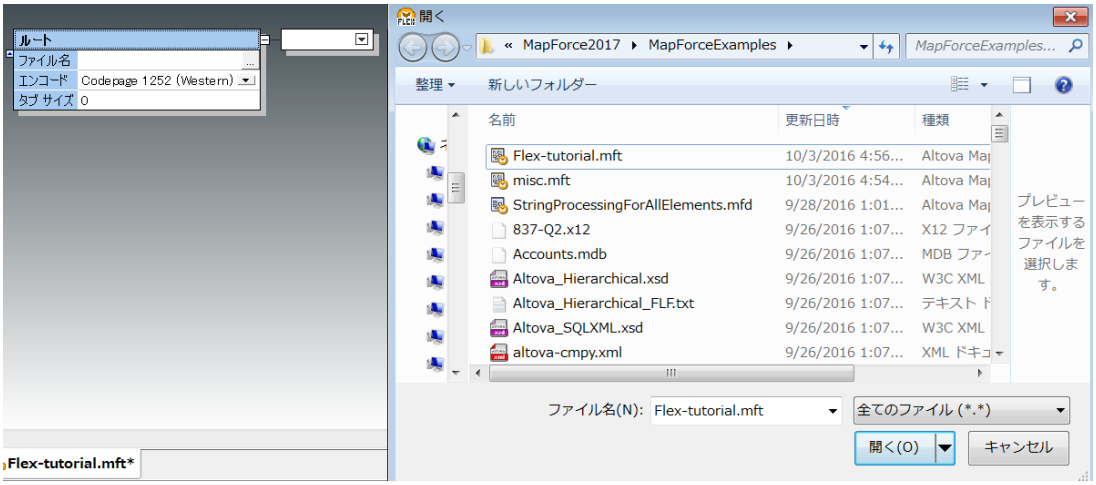
7.4.2.1 ステップ 1:FlexText テンプレートの作成

FlexText テンプレートを作成するには、以下を行います :

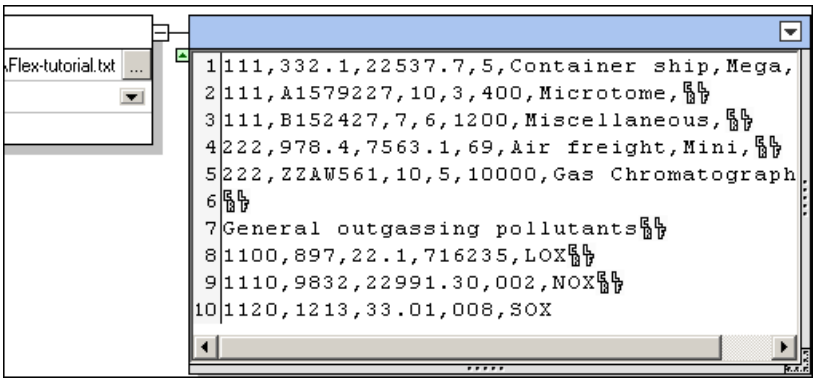
1. MapForce を開始して、新規のマッピングファイルを開きます。
2. **挿入 | テキストファイル** を選択、または、テキスト ファイルの挿入アイコン をクリックします。
3. **新しい構造タイプをグラフィカルにデザイン ...** ボタンをクリックします。



4. FlexText テンプレートのための名前を入力し、「保存」をクリックして継続します (例 Flex-tutorial.mft)。空のデザインが、「開く」ダイアログボックスと共に表示されます。



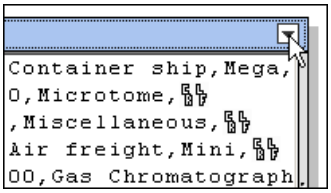
5. **...\MapForceExamples\Tutorial** フォルダー内の**Flex-tutorial.txt** ファイルを選択し、 開く をクリックして確認します。



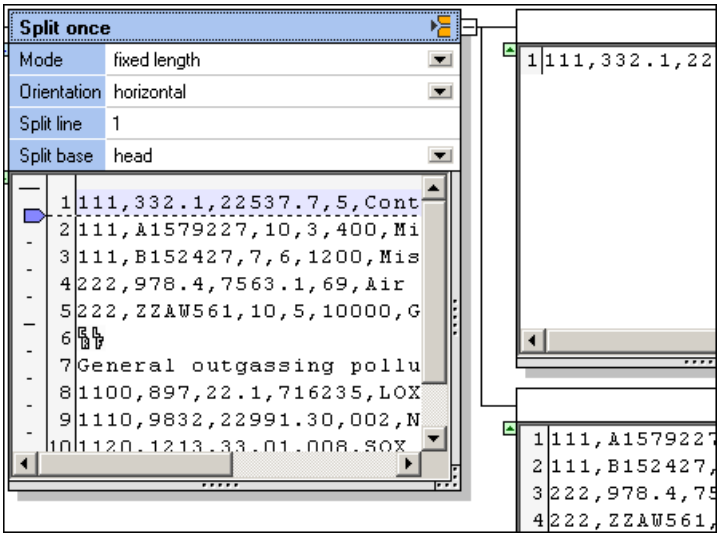
テキストファイルコンテンツが表示されます。

「デザインビューのノードテキスト」アイコン をクリックすると、アクティブなコンテナのコンテンツがサンプルテキストペイン内に表示されます。

「選択されていないノードテキストを自動的に折りたたむ」 を有効化すると、アクティブなコンテナのコンテンツが表示され、コンテンツを含む他のコンテナのコンテンツは折りたたまれます。



6. 右上のコンテナアイコンをクリックし、ポップアップメニューから **「1回分割する」** を選択します。
「1回分割する」コンテナの横に2つのコンテナが表示されます。「1回分割する」条件に関する詳細は、次を参照してください：「1回分割する」。



「1回分割する」コンテナのデフォルトの設定が表示されます：固定長、水平、および、分割する行＝1．

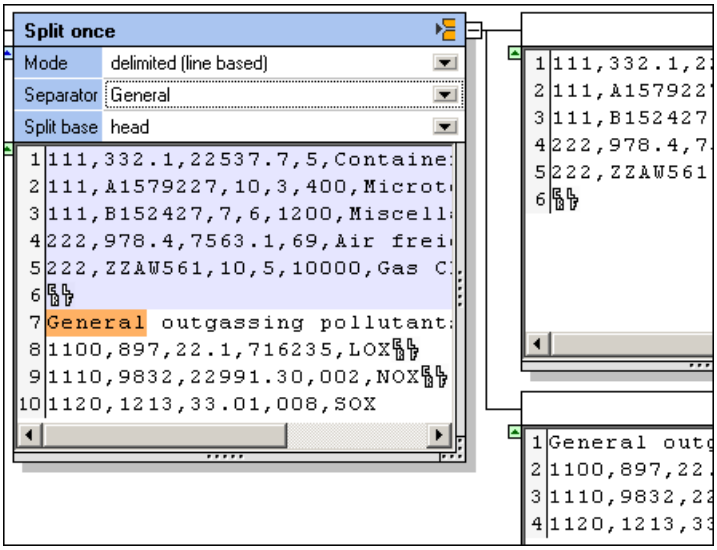
これらのデフォルトの設定の結果も表示されます：

- 上のコンテナは、テキストファイルの最初のラインを含んでおり、**「1回分割する」**コンテナ内でハイライトされています。
- 下のコンテナは、テキストファイルの残りを含んでいます。

7.4.2.2 ステップ 2:分割の条件の定義

分割の条件により、テキストのフラグメントを色々な方法で分割することができます。分割の条件を定義するには以下を行います：

1. **モード**コンボボックスをクリックして、「区切られた(行ベース)」を選択します。
2. セパレーター フィールドをダブルクリックして、「General」を入力します。

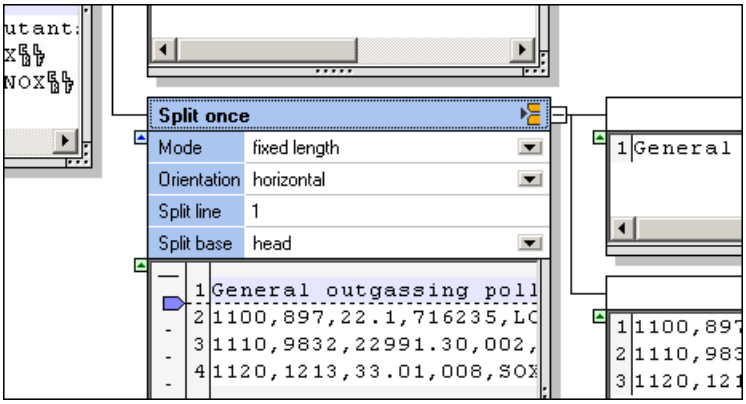


対応するコンテナ内のフラグメントが変更されます。

「General」を入力し、区切られた(行ベース)を使用して、文字列「General」を含むテキストのセクションを下のコンテナに分割することができます。セパレーターまでのテキストフラグメントは、上のコンテナ内に置かれます。

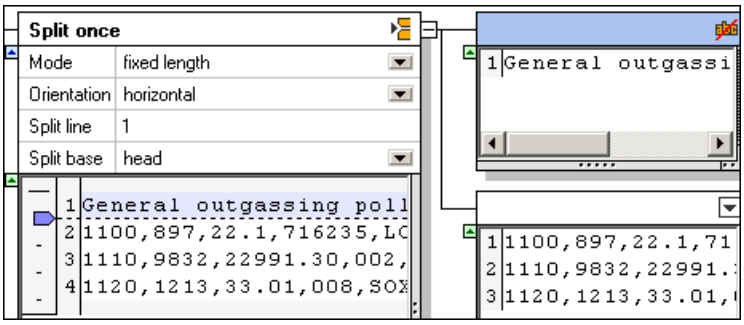
下のコンテナと作業して、1100以上の数値を持つレコードを含むCSV ファイルを生成します。

3. 下のコンテナをクリックして、**「1回分割する」**に変更します。



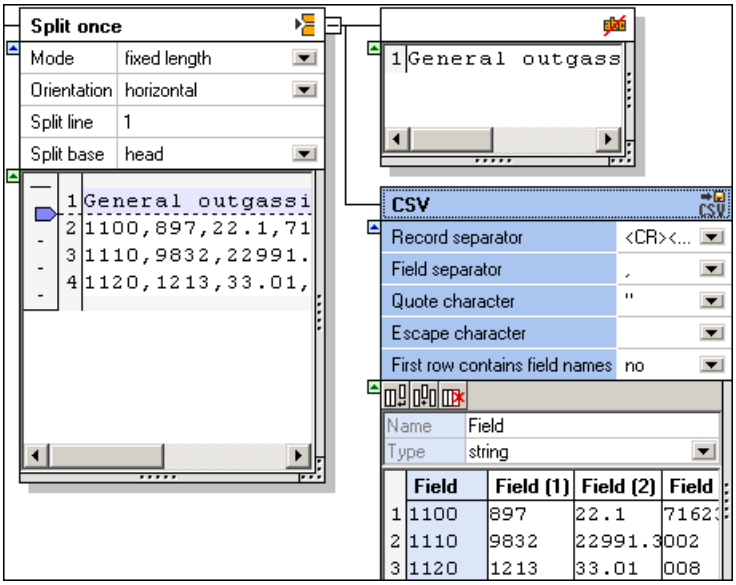
2つの新規のコンテナが作成されます。このテキストフラグメントの最初の行を分割するため、デフォルトの設定はそのままにし、無視することができます。残りのフラグメントin 下のコンテナ内の残りのフラグメントがCSV ファイルになります。

4. 上のコンテナをクリックして、**無視」**に変更します。



このコンテナのテキストフラグメント、すなわち、マッピングアイテムが、MapForce 内のマッピングに対して使用できなくなりました。

5. 下のコンテナアイコンをクリックして、**CSV として保管**に変更します。



コンテナは、テーブル形式でテキストフラグメントを表示しています。デフォルトの設定は、保持することができます。

CSV ファイルの構成：
フィールド名を変更する場合は、テーブル内のフィールドをクリックし、**「名前」**フィールド内のエントリを変更します。このコンテナから列を追加、挿入、削除することもできます。更に詳しい情報に関しては、CSV として保管を参照してください。

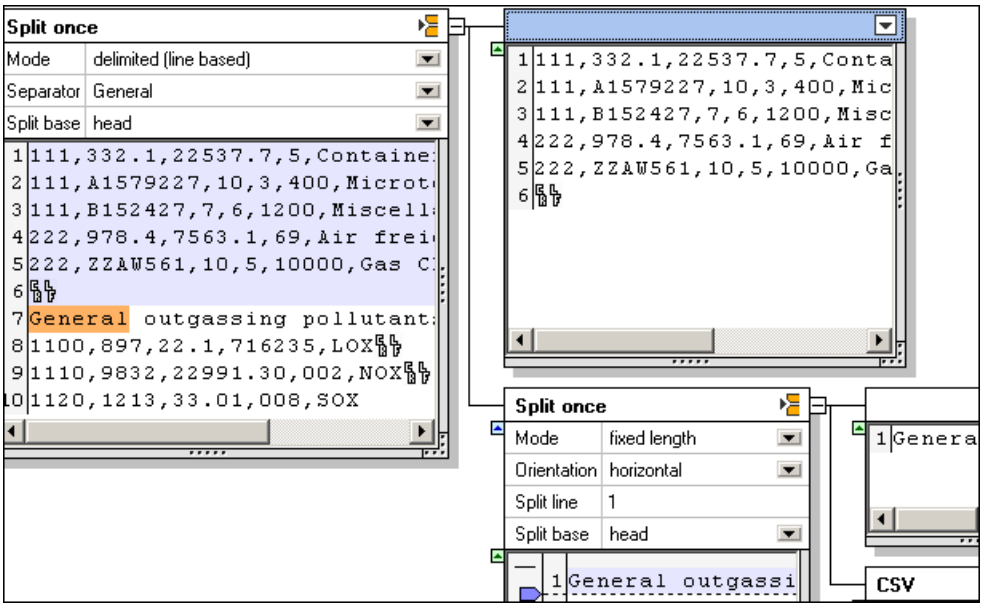
残りの テキストフラグメントの定義を続けます。

7.4.2.3 ステップ 3:各コンテナーのための複数の条件の定義

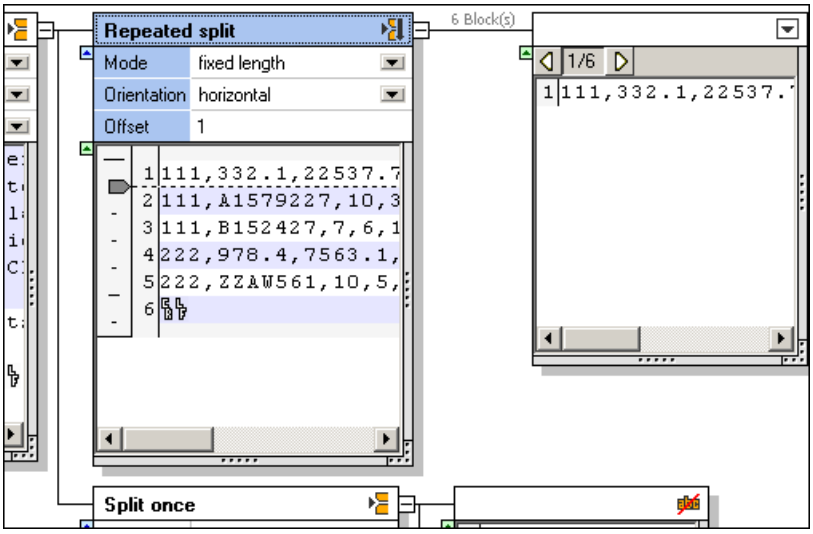
FlexText により切り替え コンテナを使用して、各テキストフラグメントに対して複数の条件を定義することができます。関連したコンテナは、定義された各条件に対して与えられます。

この時点でのチュートリアルの現状は、最初の**「1回分割する」**コンテナの下のテキストフラグメントが以下の条件で定義されています：

- 「1回分割する」コンテナは、最初の行を分割し、コンテナを無視します。
- 残りの セグメントは、CSV ファイルとして定義 /保管されます。

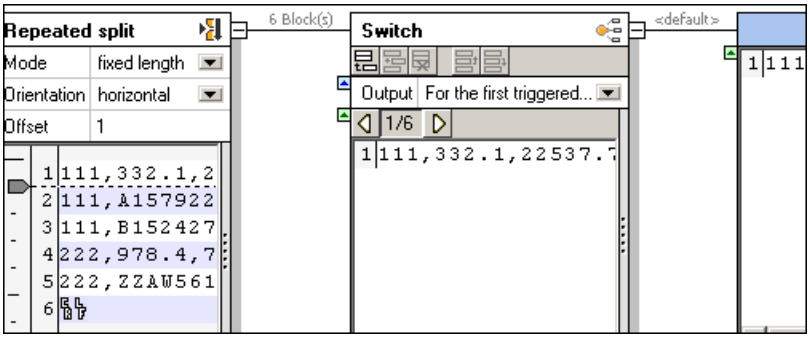


1. 上のコンテナ アイコンをクリックして、**繰り返される分割**に変更します。



現時点では、デフォルトの設定が必要とされ使用されます。テキストフラグメントは、それぞれが1行からなる複数のテキストブロックに分割されます。関連したコンテナは、各テキストブロックのプレビューを表示します。
次のテキストブロックアイコン をクリックすると、6つある全てのテキストフラグメントを順番に確認することができます。

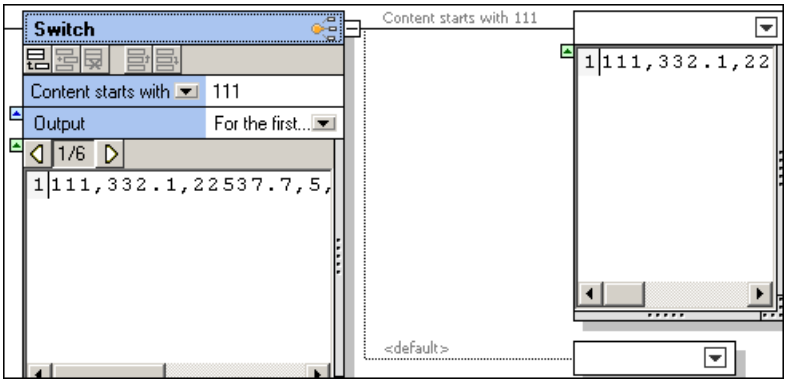
2. コンテナをクリックして、**「切り替え」**に変更します。



「切り替え」コンテナの最初の状態は、上に示されています。

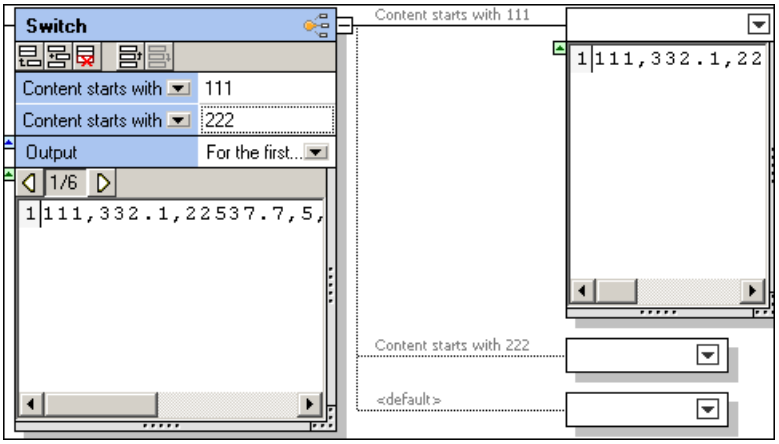
- 関連したコンテナ「デフォルト (default)」が追加されました。
- 最初のレコード 1/6 のコンテンツは、デフォルトのコンテナ内に表示されています。

3. 「切り替え」タイトルバー内の条件を追加アイコン をクリックして新規の条件を追加します。
4. 「先頭」内のフィールドをダブルクリックして、111を入力します。



これにより最初の条件が定義されます。(コンテンツが 111から開始する)関連したコンテナ 「デフォルト(default)」コンテナの上に追加されます。

5. 追加アイコンをもう一度クリックし、 「先頭」フィールドに 「222」を入力します。

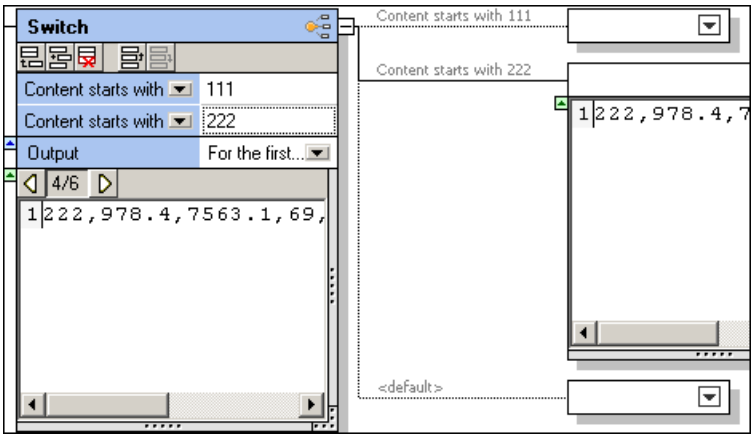


(数値 222から始まる) 3番目のコンテナが追加されました。 .

次に留意してください:

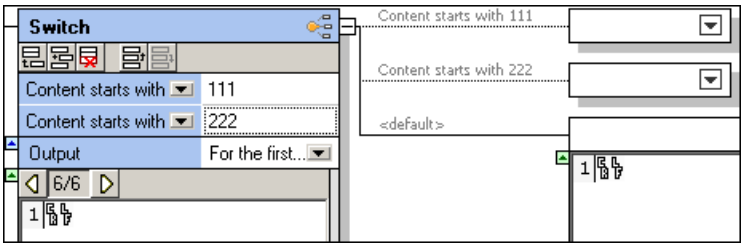
「先頭」コンボボックスをクリックすると、 「含む」オプションを選択することができます。これにより、テキストフラグメント内のどこでも発生することのできる 「String」を指定することができます。

6. 次のテキストブロックアイコン を数回クリックして、結果を確認します。



6 件中 4件目のレコードまで到達すると、コンテナ 222が開かれ、コンテンツが表示されます。

7. 6 件中の 6件目のまで到達するまでクリックし続けます。単一のCR / LF 文字 がデフォルトの コンテナに表示されます。

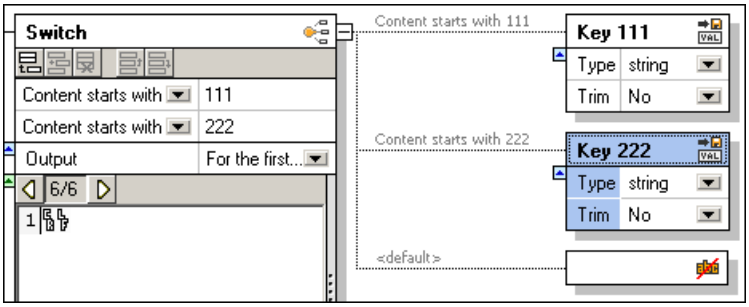


現在のブロック内のデータフラグメントが条件を満たす場合、そのブロックの **「完全なデータ」** が関連するコンテナにパスされます。条件を満たす場合、データは分割されず、関連するコンテナ、または、デフォルトのコンテナにルートされます。

8. 最初の2つのコンテナをクリックして、 **「値として保管する」** に変更します。 最後のコンテナをクリックして、 **「無視」** に

変更します。

9. 「保管」テキストをダブルクリックして、Key 111 とKey 222 などの説明テキストを追加します。

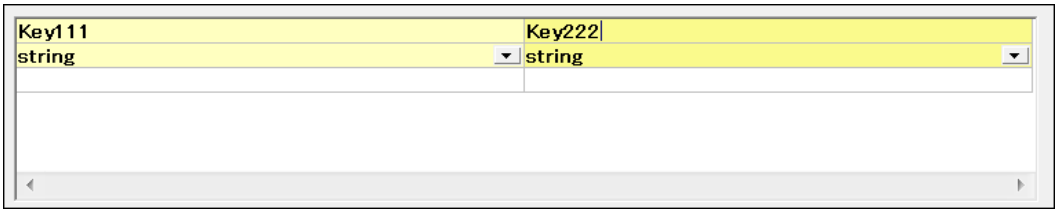


10. FlexText テンプレートを保存します。例 Flex-Tutorial.mft。

7.4.2.4 ステップ 4:ターゲットMapForce コンポーネントの作成

FlexText ソースで使用するためにターゲットコンポーネントを作成するには以下を行います :

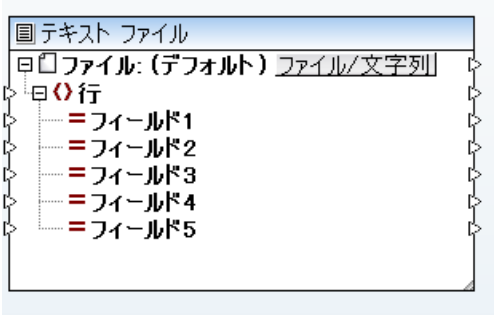
1. **挿入 | テキストファイル**」を選択、または、テキスト ファイルの挿入アイコン  をクリックします。
2. **単純な処理を使用 ..**」ラジオボタンをクリックして、**続ける**」をクリックします。
これにより、コンポーネント設定ダイアログボックスが開かれます。
3. **フィールドを追加**」ボタンをクリックして新しいフィールドを追加します。
4. フィールド1 フィールド名をダブルクリックして、**Key 111**」に変更します。



5. 他のフィールドも同様に処理し、**Key 222** と名前を付け、「OK」をクリックして確認します。
2つのフィールドのあるテキストコンポーネントが作成されます。



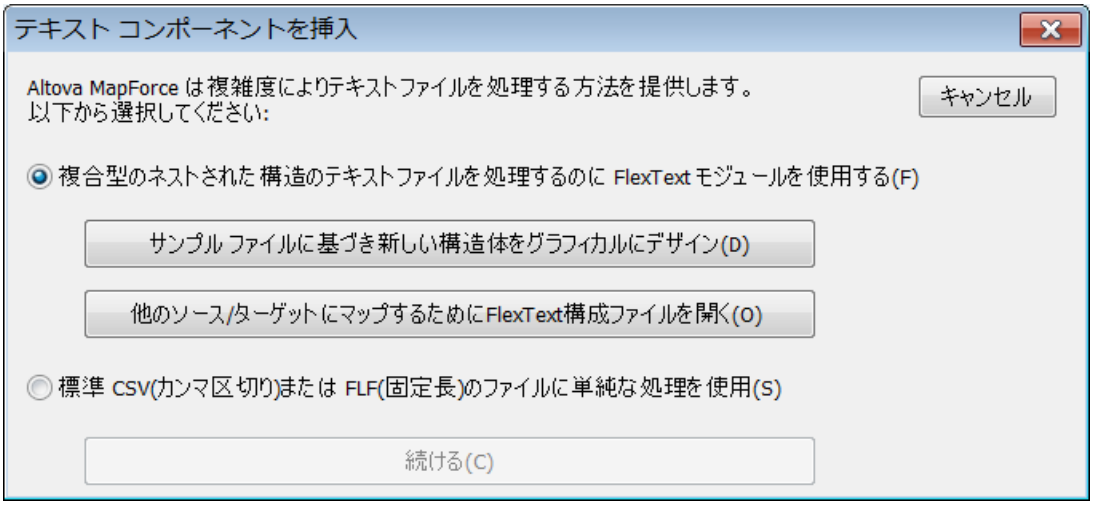
6. 同じ方法を使用して、5つのフィールドから構成される2番目のテキストコンポーネントを作成します。



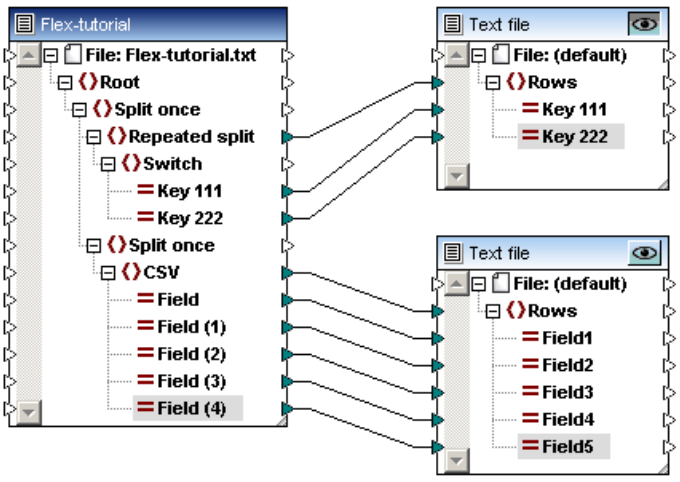
7.4.2.5 ステップ 5:MapForce 内で FlexText テンプレートを使用する

MapForce 内でFlexText テンプレートを使用する方法 :

1. アプリケーションを開始、または、MapForce に切り替え、 **挿入 | テキストファイル** を選択します。



2. **既存のFlexText 構成ファイルを開く...** ボタンをクリックして、以前に定義されているFlexText テンプレート (**Flex-tutorial.mft**) を選択します。
MapForce コンポーネントの構造は、FlexText 内のデザインビュー内のコンテナを反映しています。



3. 既に定義済みの ターゲット コンポーネントに多種のアイテムをマップし、出力タブをクリックして、結果をプレビューします。

上のテキストコンポーネントのマッピングプレビュー：

```
"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,,
"
```

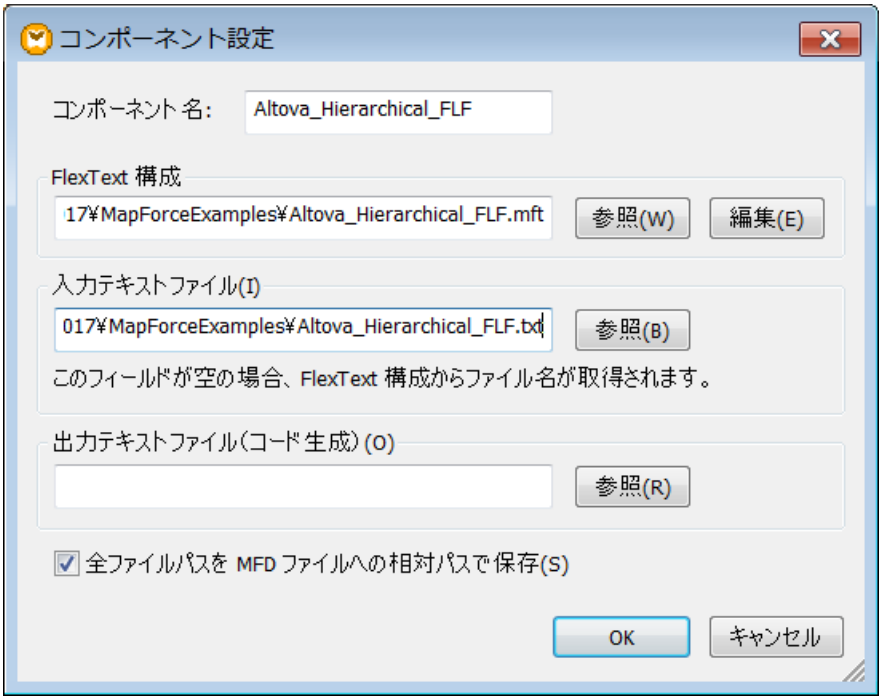
下のテキストコンポーネントのマッピングプレビュー：

```
1100,897,22.1,716235,LOX
1110,9832,22991.30,002,NOX
1120,1213,33.01,008,SOX
```

7.4.3 FlexText コンポーネント設定

FlexText コンポーネントをマッピングエリアに追加すると、コンポーネント設定ダイアログボックスから、適用することのできる設定を構成することが可能になります。コンポーネント設定ダイアログボックスを以下の方法を用いて開くことができます：

- **「コンポーネント」** メニューから、 **「プロパティ」** をクリックします (このメニューアイテムは、コンポーネントを選択すると有効化されます)。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。



FlexText コンポーネント設定ダイアログボックス

使用することのできる設定は以下の通りです。

コンポーネント名	コンポーネント名は、コンポーネントを作成すると自動的に生成されます。名前はいつでも変更することができます。 コンポーネント名が自動的に生成され、その後、インスタンスファイルを選択すると MapForce は任意でコンポーネント名を更新するようにプロンプトします。 コンポーネント名には、スペースと終止符文字を含むことができます。コンポーネント名には、スラッシュ、バックスラッシュ、コロン、二重引用符、先頭および末尾のスペースを含むことができません。一般的には、コンポーネント名を変更する際、以下の点に注意してください: • FlowForce Server にマッピングをデプロイする場合、コンポーネント名を一意の名前にしてください。 • コマンドラインに入力することのできる文字のみを使用することが奨励されます。国別使用文字によっては Windows とコマンドライン内で異なるエンコードを使用する場合があります。
FlexText 構成	コンポーネントにより使用されている FlexText テンプレート (mft) の名前、または、パスを指定します。 新規のテンプレートを選択するには、**参照...** をクリックします。FlexText 内のテンプレートを編集するには、**編集...** をクリックします。
入力テキストファイル	MapForce がデータを読み取るテキストファイルを指定します。このフィールドは、ソースコンポーネントにとり重要で、最初にコンポーネント作成した際に作成され、テキストファイルに割り当てられます。 ソース テキスト ファイルは、FlexText テンプレート内で定義されているものを優

	先します。FlexText テンプレート内で定義されているファイル名がソースとして使用されます。
出力テキストファイル (コード生成のため)	MapForce がデータを書き込むための テキストファイルのパス名を指定します。このフィールドは、コードを生成する際に、ターゲットコンポーネントにとって重要です。フルパスを入力することにより、ターゲットディレクトリを特定的に定義することができます。例えば、**c:\myfiles\sequence.txt** などが挙げられます。
MFD ファイルに相対的な全てのファイルパスを保存する	このオプションが有効化されていると、MapForce は、コンポーネント設定ダイアログボックスに表示されている MapForce Design (.mfd) ファイルの場所に対して相対的なファイルパスを保存します。

7.4.4 ターゲットコンポーネントとして FlexText を使用する

FlexText コンポーネントの主な使用方法は、テキストファイルのセクションを削減し、分割し、FlexText コンポーネントの対応するアイテムを他の ターゲット コンポーネントにマップすることです。FlexText を、しかしながら、ターゲット コンポーネントとして使用し、個別のファイルを構成、または、再構成して、単一の ファイルに構成することができます。期待する結果を達成するには、試行錯誤を必要とする可能性があります。

FlexText コンポーネントを**ターゲット**として使用すると、以前に定義された操作が元に戻されます。ファイルを多数のサブセクションに分割する代わりに、ファイルを構成、または、再構成します。

一般的には、ターゲットとして使用される場合、 FlexText テンプレート内で定義されている各操作を元 (ボトムアップ) に戻します、 :

- 「分割」は、 「マージ」になります。例 繰り返される分割のマッピングは、 「」 により区切られ、マージ アイテムは 「」により分割されます。
- 「保管」は、 「ロード」になります。
- 「切り替え」は、 最初の一致項目を選択する」になります。

ターゲット コンポーネントとして FlexText を使用する場合、以下に注意してください :

- データソース コンポーネントと FlexText コンポーネントの入力 アイテム 1つの間の接続が構築されると、 FlexText コンポーネント データソースは、無視されます。データは、新規にマップされたソースコンポーネントにより与えられ、優先順位を有します。
- テキストは、 (CSV とFLF として保管) 「保管 ..」 コンテナにマップされ、セパレーターは保持されます。しかしながら、 「保管 .」 ノードの上のノード内で固定長 分割が起こる場合、テキストは切り捨てられる可能性があります。
- 固定された幅の分割では、分割ベース=ヘッドの場合、左 /上のセクションが定義されている長さに対し切り捨てられる可能性があります。または、分割ベース=テールの場合、右 /下 セクションが定義されている長さに対し切り捨てられる場合があります。テキストが短すぎる場合、スペース文字が、セクションに挿入されます。
- FlexText は、通常、分割 操作のアイテムの間にセパレーター (固定された分割のための空白スペース) を挿入します。しかし、これは、 「区切られた (行ベース)」 分割には、適応されません。 「区切られた (行ベース)」 操作は、操作を完全に元に戻すことはできません。 「区切られた」 テキストは、最初の行内で発生し、テキストに含まれます、このための、自動的な処理により正確に追加することはできない可能性があります。
  - o 区切られた (行ベース) は、最初の行が不在の場合セパレーターは追加されません。
  - o 区切られた (フロート) は、 2つのセクション間にセパレーターを追加します。
- 切り替え 操作は、簡単なケース以外は元に戻すことはできません。このデータ切り替えは、自身のブランチをデータが含まれる最初のブランチのためにスキャンし、このデータを使用、または、挿入します。切り替え操作の最初の接続のみがマップされます。 残りの切り替えコンテナにデータを転送するには、フィルターを残りの接続のために定義し、繰り返される分割アイテムに入力されるように、切り替えアイテムが単一アイテムを返すように切り替え親アイテムの複製を作成する必要があります。
- 単一の分割 コンテナの子に対してマッピングすると、最後のアイテム以外すべてのマッピングの結果は破棄されます。複数の結果が生成された場合でも、単一の結果が時保持されます。

次のXML スキーマ コンテンツモデルの例は、ターゲットとして使用されているFlexText の振る舞いについて説明しています：

- 繰り返される分割は、繰り返し要素です。
- 単一の分割は、 シーケンス'コンテンツモデル グループの一部を作成します。
- 保管は choice' コンテンツモデル グループを作成します。各ケースが可能な子要素です。
- 保管は、単純型の要素を作成します。

7.4.5 FlexText レファレンス

レファレンスのセクションは、FlexText の異なる機能について説明し、特定の結果を取得するための使用方法示しています。

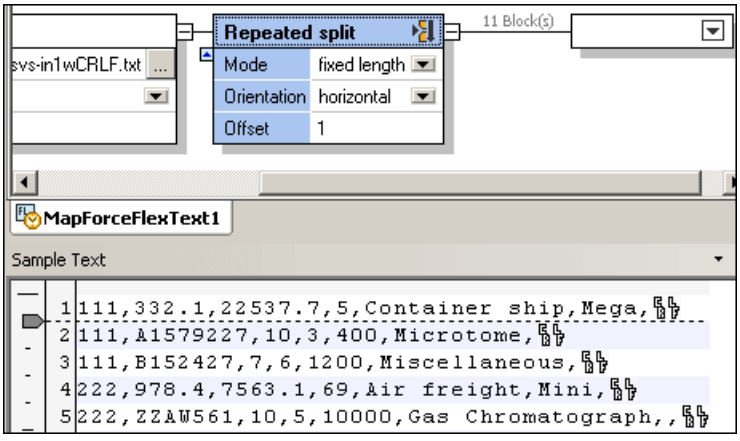
7.4.5.1 繰り返される分割



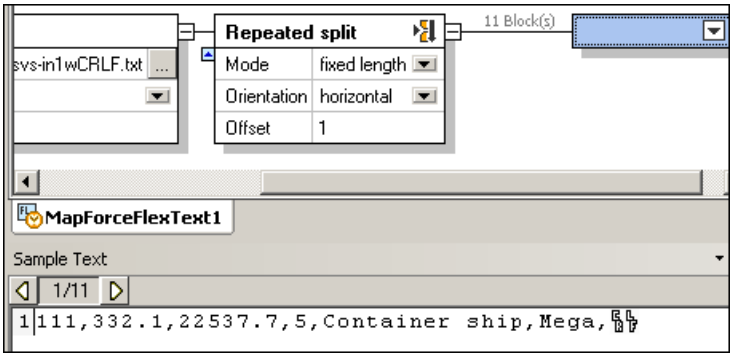
このオプションを使用すると、単一の コンテナが最初に作成されます。コンテナには、**繰り返される分割**の条件のセットにより定義されたテキストが含まれます。繰り返される分割のバージョンは以下の通りです：固定長、区切られた フロート、区切られた行ベース と区切られた行の開始 ..。

このオプションを最初に選択すると、デフォルトのパラメーターは、自動的 2設定され、結果 フラグメントは、関連したコンテナに表示されます。繰り返される分割 コンテナが現在アクティブであり、サンプルテキストペイン内にプレビューは、すべての現在のレコード列を表示することに注意してください。

コンテナのデフォルトの設定：
モード 固定長
方向 水平
オフセット 1



デフォルトの結果：
テキストの各行は、新しいコンテナ内の行 /レコードとして表示され、オフセットは1です。新しい コンテナをクリックして、コンテンツをプレビューします。 サンプル テキスト **「スクロールの矢印」**によりこの設定により生成された 11件のブロック/フラグメントのそれぞれをスクロールすることができます。



モード -固定長

繰り返される分割 (固定長)モードを使用して、テキストを複数の水平、または、垂直なフラグメントに分割することができます。**繰り返される分割 (固定長)**モードに適用することのできる設定は、以下で説明されています。

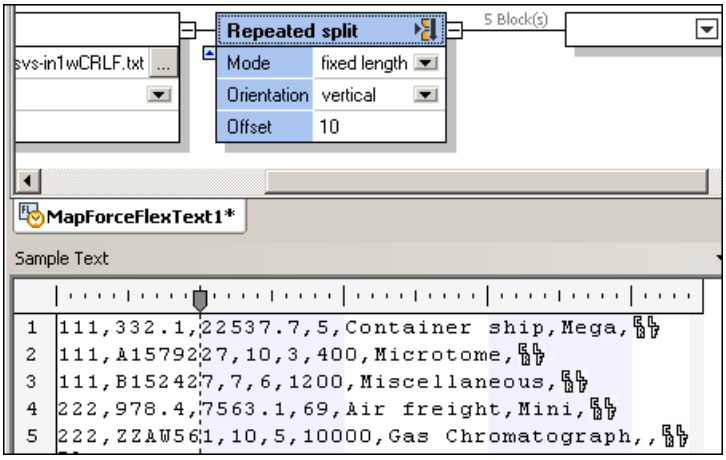
方向

テキストフラグメントが水平方向、または、垂直方向に分割されるかを指定します。フラグメントを複数の水平のセクションに分割する場合は、「水平」を選択します。フラグメントを複数の垂直のセクションに分割する場合は、「垂直」を選択します。

オフセット

方向が「水平」に設定されている場合、これにより設定は、各フラグメントが含む列数を指定します。方向が「垂直」に設定されている場合、この設定は、各フラグメント内の幅を指定します。デフォルトのオフセットは、「1」です。この設定を変更するには、以下のうちの1つをおこなってください:

- オフセット フィールドに値を挿入します。
- タブを垂直、または、水平ルーラーにドラッグします。



モード -区切られた (フロート)

区切られた (フロート)を使用した繰り返される分割」モードを以下のシチュエーションで使用することができます :

- 指定されたセパレーター文字が結果フラグメントから取り除く。
- セパレーターがインラインの箇所のテキストを分割する (例えば、CR/LF 文字を含まないテキスト)。

メモ: フラグメントは、セパレーターの後の最初の文字と同じセパレーターの次のインスタンスの最後の文字の前の間として定義されます。下の例で示されるように、このルールに対しての例外は、最初を最後のフラグメントです。

区切られた (フロート)を使用した繰り返される分割」)モードに適用することのできる設定は以下にリストされています。

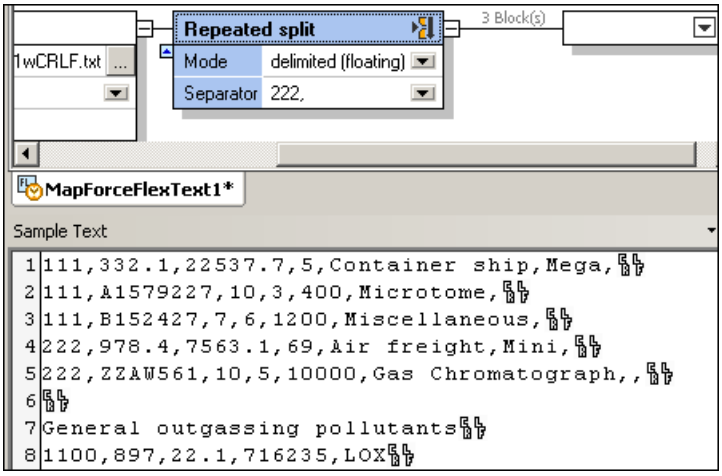
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。 (次を参照 :正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

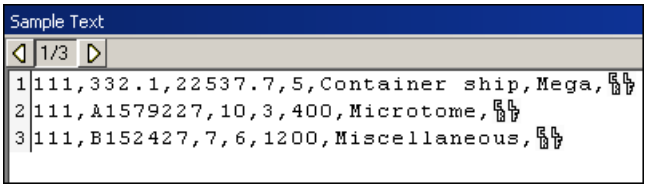
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は (セパレーター無し) 無し」です。

例えば、セパレーター 「222」 を下に表示されるテキストに対して使用すると、3つの個別のフラグメントが作成されます。



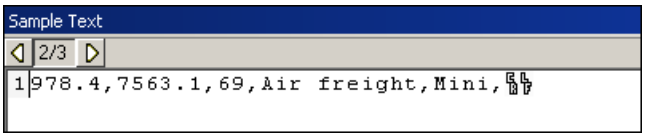
最初ののフラグメントは、フラグメントの開始から最初のセパレーター (「222」)の開始、すなわち「111」から「Miscellaneous」まで、の全ての文字 を含みます。



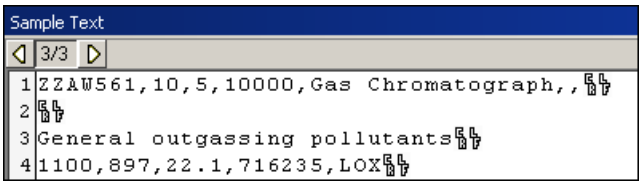
セパレーターが、フラグメント内の最初の行の文字の最初のセットではない場合、このサンプルで示されるように、最初ののフラグメントには、セパレーターの最初のインスタンスまでの全てのテキストが含まれます (例えば、 「222」)。

「111」がセパレーターの場合、最初ののフラグメントは、ソースフラグメントの最初の行の最初に現れるセパレーターである、ゼロの長さの文字列です。

2番目のフラグメントは、セパレーター 222 を含む最初の行を、セパレーター自身は無しで含みます。



3番目のフラグメントは、セパレーター 222 を含む次の行を、セパレーター自身は無しで、テキストファイル/フラグメントの終わりまで含みます。



モード -区切られた（行ベース）

「区切られた（行ベース）を使用した繰り返される分割」モードは、次のように振る舞い、テキストを複数の フラグメントに分割します：

- このモードは、セパレーター フィールドに入力したセパレーター文字により定義された複数のフラグメントを作成します。
- セパレーター文字は、フラグメント内に含まれています。
- フラグメントがセパレーターを含む最初の行までのセパレーターを含む全体の行として定義されます。
- 最初の行でセパレーターが発生しない場合、最初ののフラグメントには、セパレーターを含む最初の行までの行が含まれます。

「区切られた（行ベース）を使用した繰り返される分割」モードに適用することのできる設定は以下にリストされています。

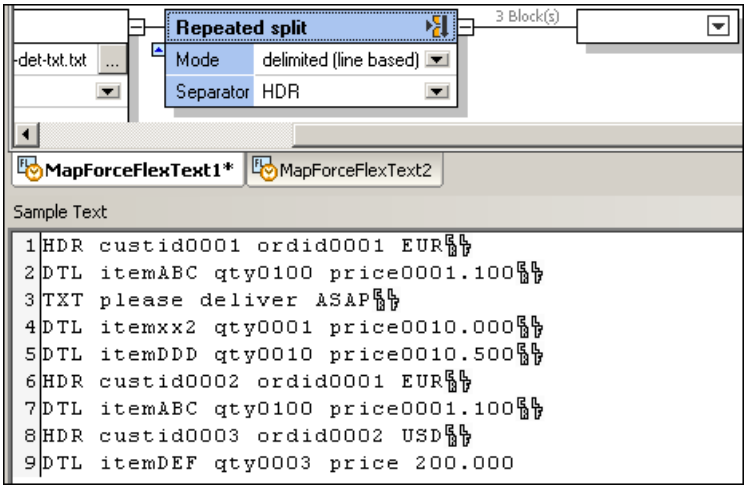
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。（次を参照：正規表現を使用してテキストを分割する）。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」（セパレーター無し）。

例えば、セパレーター **「HDR」**をテキストに対して下に表示されるように使用すると、3つの個別の フラグメントが作成されます。



最初ののフラグメントは、全ての行から同じセパレーターを含む次の上までの ファイル/フラグメントから開始する全ての文字を含んでいます。

```
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
```

このオプションにより、2つのセパレーターの間の無数の行にアクセスすることが許可されます。これは、(このサンプルでは、「DTL」または「TXT」などの)任意の、または、シーケンスではない、レコード型をもつファイル内で役に立ちます。

2番目のフラグメントには、2番目の「HDR」の発生から次の発生までのすべてのテキストが含まれています。

```
1 HDR custid0002 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
```

3番目のフラグメントには、3番目の「HDR」の発生からの全てのテキストが含まれています。

```
1 HDR custid0003 ordid0002 USD
2 DTL itemDEF qty0003 price 200.000
```

モード -区切られた 行の開始)

「区切られた 行の開始 を使用した繰り返される分割」モードは、次のように振る舞い、テキストを複数の フラグメントに分割します:

- このモードは、セパレーター フィールド内に入力するセパレーター文字により定義される複数の フラグメントを作成します。
- セパレーター文字 は、フラグメント内に含まれています。
- フラグメント、セパレーターから開始して、行の最初に同じセパレーターが含まれる次の行までの全体の行として定義されます。
- セパレーターが、最初に行に存在しない場合、セパレーターを含む最初の行までの行を含んでいます。

「区切られた (行の開始) を使用した繰り返される分割」モードに適用することのできる設定は以下にリストされています。

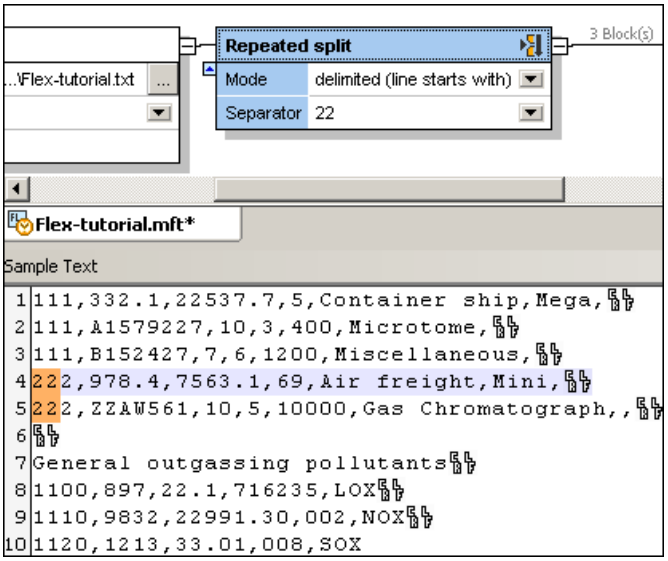
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。 (次を参照 : 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

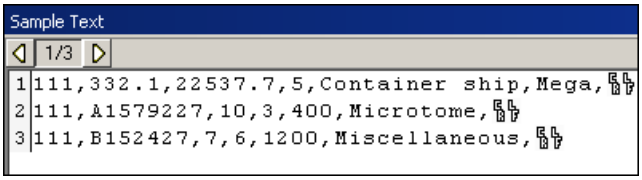
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」(セパレーター無し)。

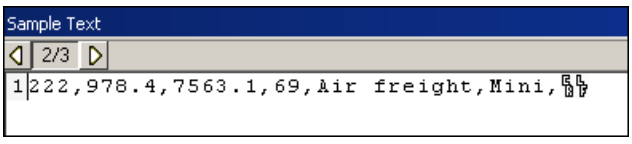
例えば、セパレーター「22」を下のテキストに対して使用すると、3つの個別のフラグメントが作成されます :



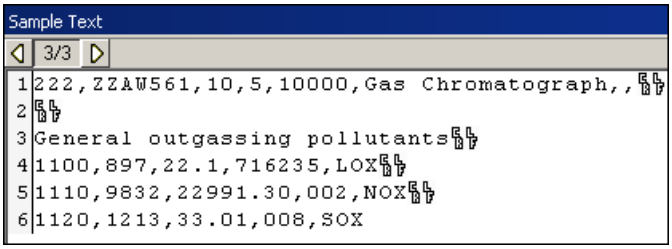
最初のフラグメントでは、セパレーター「22」を含む行までの全ての行を含む、ファイル/フラグメントから開始する全ての文字が含まれています。



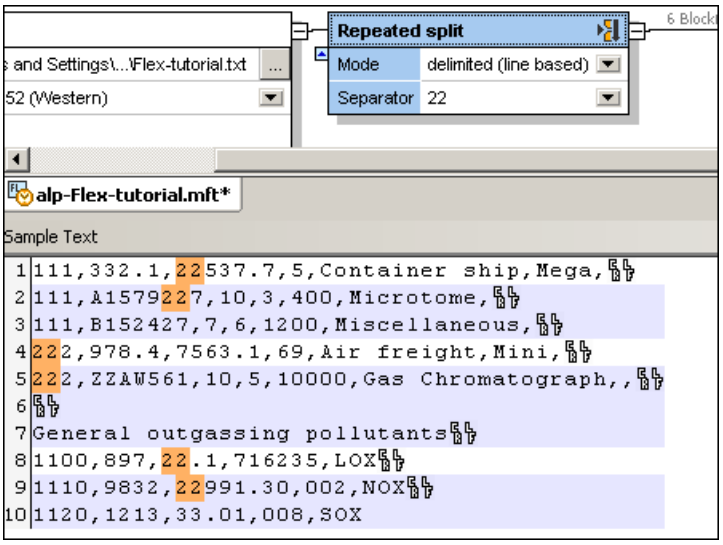
2番目のフラグメントでは、「22」の2番目の出現からのすべての文字/列から、の次の出現までが含まれています。この場合、は1行のみが条件を満たします。



3番目のフラグメントでは、「22」の3番目の出現から終わりまでのすべての文字/列が含まれます。



比較のために、**区切られた (行ベース)**モードとセパレーター 「22」としてセパレーターが使用された場合、の結果は以下のとおりです：



行内に 22 を含む行により構成される、6件のフラグメントが存在します。

7.4.5.2 1回分割する

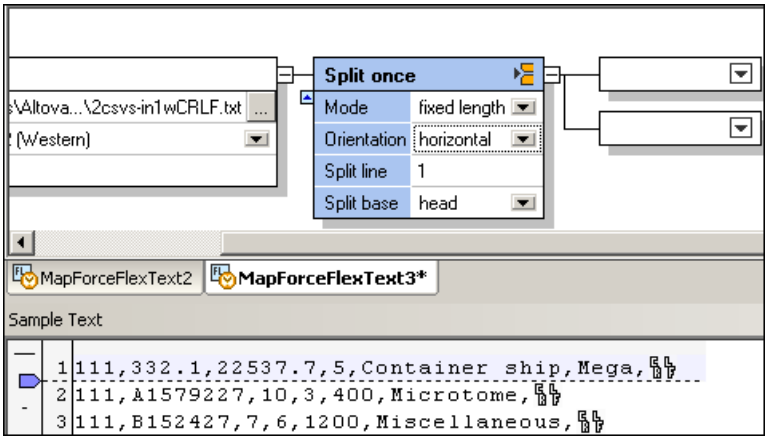


このオプションを使用すると、2つの上下に並ぶコンテナが作成されます。上のコンテナには、**「1回分割する」**コンテナ内で設定された条件のセットにより設定されているテキストが含まれています。下のコンテナには、テキストファイル/フラグメントの残りが含まれています。「1回分割する」オプションには次のバージョンが存在します：固定長、区切られたフロート、と区切られた行ベース。

このオプションを最初に選択すると、デフォルトのパラメーターが自動的に設定され、結果のフラグメントは両方のコンテナに表示されます。**「1回分割する」**コンテナが現在アクティブであり、すべての現在のレコード/列のプレビューをサンプルテキストペイン内に、表示することに注意してください。

コンテナのデフォルトの設定：

モード	固定長
方向	水平
分割する行	1
分割ベース	ヘッド



デフォルトの結果：
テキストの最初の行は、上のコンテナ内に表示されます。下のコンテナは、テキストファイル/フラグメントの残りを含んでいます。

モード - 固定長

テキストの始め、または、終わりに相対的なある特定の行、または、列で **「回分割する」 (固定長)** モードを使用して、水平 または、垂直な フラグメントに分割することができます。 . **「回分割する」 (固定長)** モードに適用することのできる設定は、以下で説明されています。

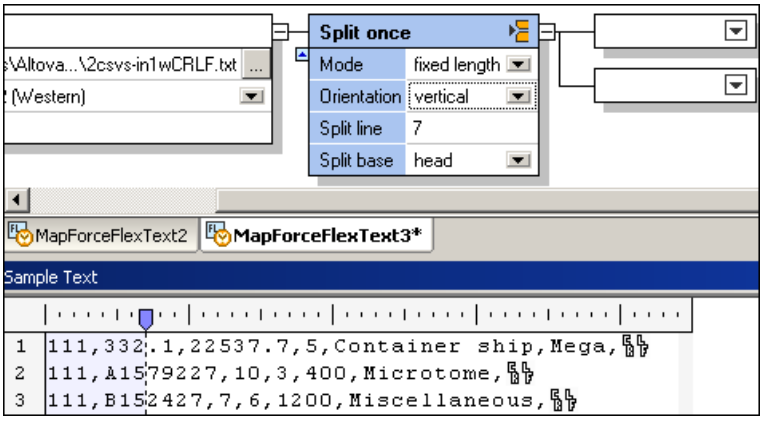
方向

テキストフラグメントが水平方向、または、垂直方向に分割されるかを指定します。フラグメントを複数の垂直のセクションに分割する場合は、 垂直」を選択します。デフォルトの方向は、 水平」です。

分割する行

フラグメントが2つに分割された後の行数 (または列数) を指定します。デフォルトの オフセットは、 「1」 です。この設定を変更するには、以下のうちの1つをおこなってください：

- **分割する行** フィールドに値を挿入します。
- タブを垂直、または、水平ルーラーにドラッグします。



分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

モード -区切られた (フロート)

「区切られた (フロート)モードを使用した1回分割」モードを使用して、カスタムセパレーターを用い、テキストを2つのフラグメントに分割します。これは、通常、CR、または、LF 文字を含まないファイルに対して使用し、特定の行内の場所でフラグメントを2つに分割する場合役に立ちます。以下に注意してください:

- このモードは、セパレーター フィールドに入力する、セパレーター文字により定義される2つのフラグメントを作成します。
- セパレーター文字 は、フラグメント内に含まれていません。
- 最初ののフラグメントは、ファイル/フラグメントの最初の文字とセパレーターの前の最後の文字の間のテキストとして定義されています。
- 2番目のフラグメントは、セパレーターの後の最初の文字から、ファイル/フラグメント内の最後の文字として定義されています。
- セパレーターが、ファイル/フラグメントの最初の/最後の位置で表示される場合、 2つの結果 コンテナの最初は空のままです。

「区切られた (フロート)モードを使用した1回分割」モードに適用することのできる設定は以下にリストされています。

正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。 (次を参照 :正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は 無し」 (セパレーター無し).

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

モード -区切られた (行ベース)

「区切られた (行ベース)モードを使用した1回分割」を使用して、セパレーターが行に存在するテキストを2つのフラグメントに分割します。セパレーターを含む列は、分割されず、全体が保留されます。これは、レコード区切り文字 (CR/LF)を含むファイルに対して、役にたち、フラグメントを 2つの個別の フラグメントに分割します。以下に注意してください:

- このモードは、セパレーター フィールドに入力するセパレーター文字により定義されている2つのフラグメントを作成します。

- セパレーター文字 は、フラグメント内に含まれています。
- 最初ののフラグメントは、含まれているセパレーターまでのすべての行のテキストとして定義されています。
- 2番目のフラグメントは、ファイル/フラグメントの終わりのセパレーターを含むテキストと行として定義されています。
- 最初の/最後の行でセパレーターが出現すると、ファイル/フラグメント内に、表示されます。上のコンテナは、空のままです。

区切られた (行ベース)モードを使用した1回分割」モードに適用することのできる設定は以下にリストされています。

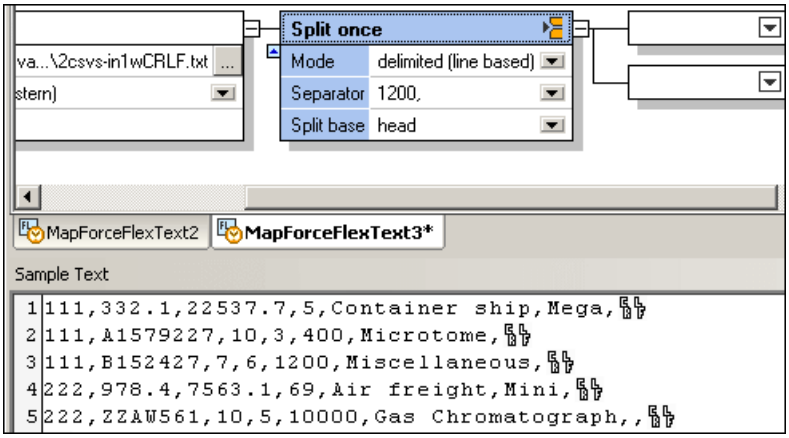
正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。 (次を参照 :正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は 無し」 (セパレーター無し).

例えば、セパレーター 「1200」 を下のテキストに対して、使用すると、2つの フラグメントが作成されます。



最初ののフラグメントには、セパレーターを含む行までが含まれます。

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
```

2番目のフラグメントには、セパレーターを含む全体の行、および、終わりまでの残りのすべての行をが含まれます。

```
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,,
```

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

モード -区切られた (行の開始)

分割が、指定されたセパレーターで開始する最初の行で発生する場合、 **区切られた (行の開始)モードを使用した1回分割** モードを使用します。 このモードを選択すると、テキストの 2つのフラグメントは以下のように作成されます :

- 最初ののフラグメントには、セパレーターが存在するまでの行の全てのテキストが含まれています。
- 2番目のフラグメントには、セパレートの存在する行を含む残りのテキストが含まれます。
- 最初の行にセパレーターが存在する場合、分割はおこなわれません (最初の結果的なフラグメントは、空のままです)。

区切られた (行の開始)モードを使用した1回分割 モードに適用することのできる設定は以下にリストされています。

正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。 (次を参照 :正規表現を使用してテキストを分割する)。デフォルトの設定は 「いいえ」です。

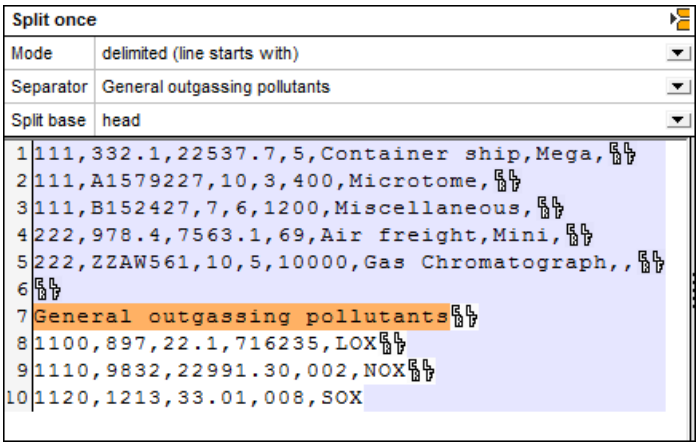
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は 無し」 (セパレーター無し).

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

例えば、下のイメージでは、指定されたセパレーター (「General outgassing pollutants」)問い定する最初の行で分割は発生します。分割ベースオプションが **テール** に設定されている場合、分割は、 (このサンプルでは同じ行に存在する)終わりから開始する指定されたセパレーターから開始する最初の行で発生します。



最初の結果的なフラグメントは、以下のとおりです :

```
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,,
```

2番目の結果的なフラグメントは、以下のとおりです：

```
General outgassing pollutants
1100,897,22.1,716235,LOX
1110,9832,22991.30,002,NOX
1120,1213,33.01,008,SOX
```

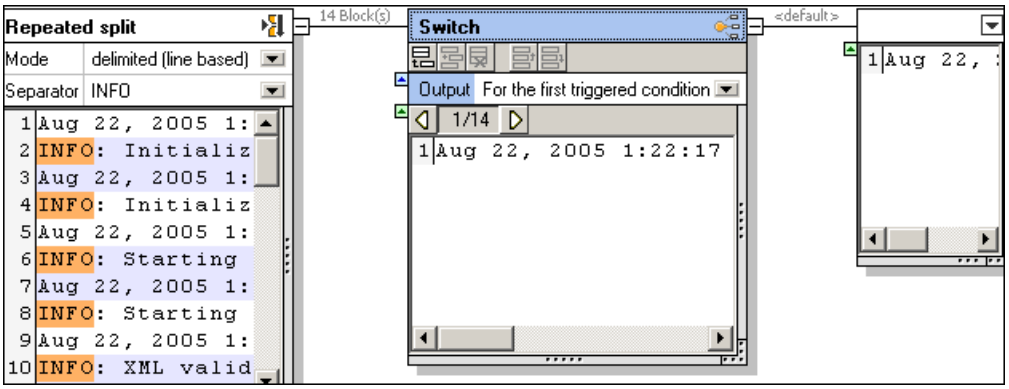
7.4.5.3 切り替え



「切り替え」オプションを使用すると、単一のテキストフラグメントのために複数のキーワード、条件を定義することができます。定義する各キーワードには、特定の条件が満たされた場合データを受け取る、自身の関連したコンテナが存在します。条件が満たされない場合、特定のフラグメントは、「デフォルト」コンテナにマップされます。

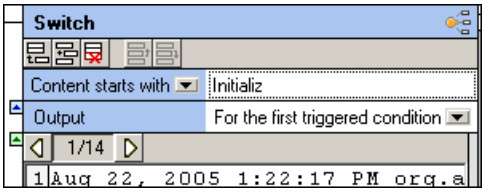
コンテナのデフォルトの設定：
出力　　最初のトリガーされた条件

下のサンプルは、個別のプロセスが分けられ、マップすることのできるTomcat ログファイルを処理します。「切り替え」コンテナを最初に定義する場合、**デフォルトの**コンテナのみが「切り替え」コンテナの右に表示されます。全てのデータが自動的にパスされます。



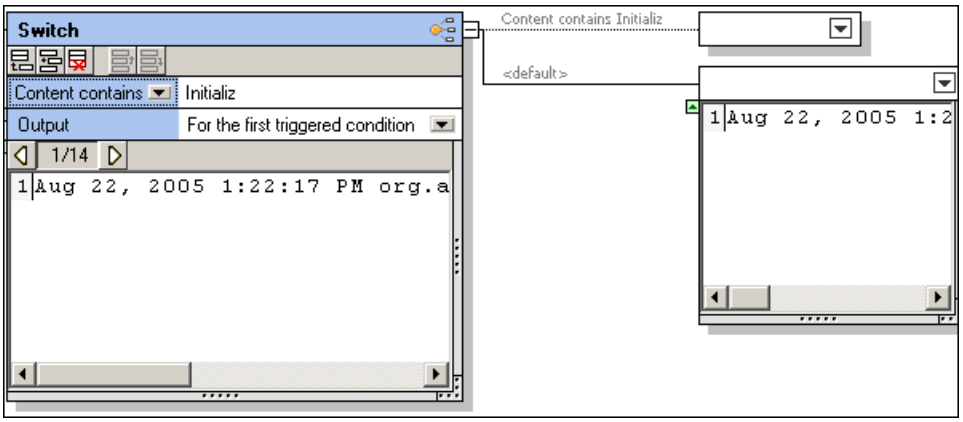
繰り返される分割コンテナでは、区切られた（行ベース）を使用して、ログファイルからすべてのINFO のセクションを分離し、「切り替え」コンテナにパスします。

1. 追加アイコン  をクリックして、新しい条件を「切り替え」コンテナに追加します。
2. 「**Content starts with**」フィールドをダブルクリックして、「**Initializ**」を入力し、Enter キーを押します。

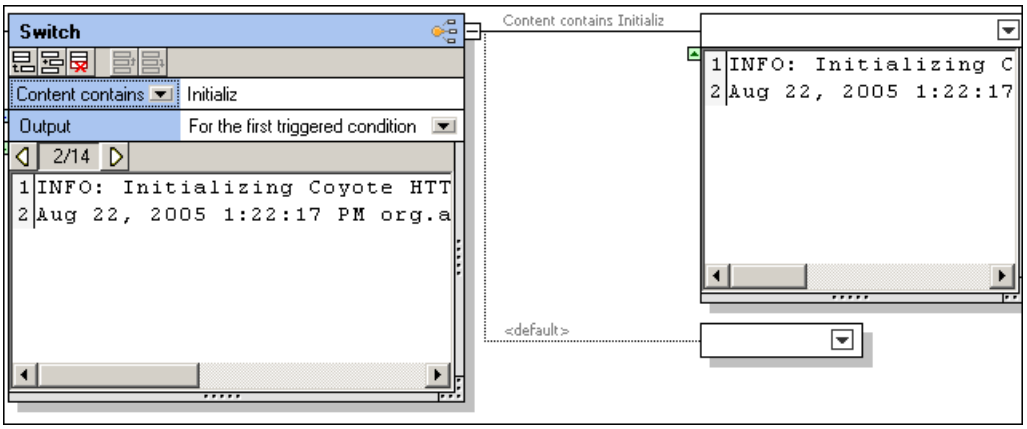


新しいコンテナが追加されます。条件が満たされる場合、データがこのコンテナに送られます。条件を満たさない場合、データはデフォルトのコンテナに送られます。

3. **先頭** コンボボックスをクリックして、「含む」に変更します。
最初の条件は、定義されており、下で結果を確認することができます。
最初のフラグメントは、「Initializ」を含みませんが、そのコンテンツは、**デフォルトの** コンテナに送られます。

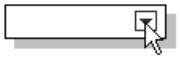


4. 次を表示するアイコン をクリックして、次のテキストフラグメントを確認します。

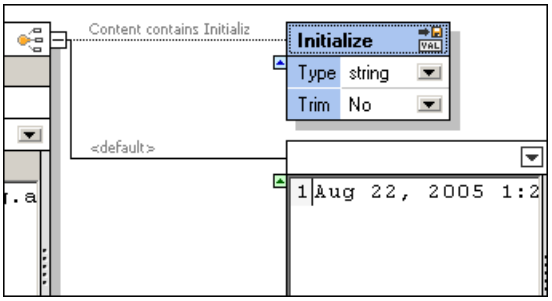


「Initializing...」フラグメントが関連したコンテナ内に表示され、デフォルトの コンテナ は空です。フラグメントをステップスルーすることにより、個別のコンテナの内容をプレビューすることができます。

5. コンテナ アイコンボタンをクリックして、 **「値として保管」** を選択します。

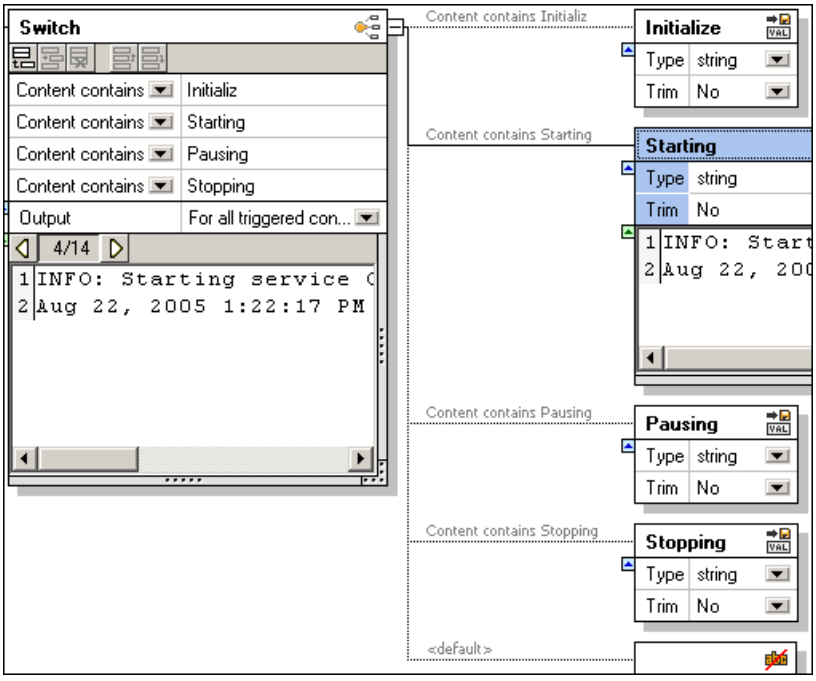


6. 保管」タイトルバー内をダブルクリックして、テキストを変更します。例 Initialize。



7. 追加アイコン をクリックして、「切り替え」コンテナに新しい条件を追加します。
8. **「先頭」** フィールドをダブルクリックして、「Starting」を入力し、Enter キーを押します。
追加することのできる条件に制限はありません。例 「Pausing」および「Stopping」。

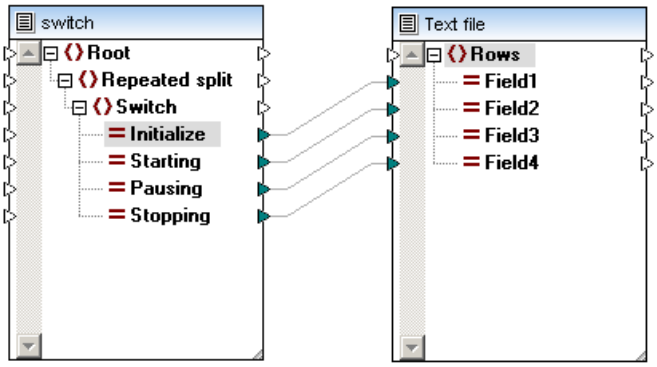
MapForce で識別しやすいように、各関連したコンテナに名前を与えます。



上のスクリーンショットは、4つすべての条件とブロック/フラグメント第 4番目の「Starting」コンテナのコンテンツを表示しています。関連したコンテナは、MapForce コンポーネント内で簡単に識別できるように、すべて名前が付け替えられています。

上へ/下へ移動ボタン、または、 を使用して、条件リスト内で条件を上下に移動することができます。

9. テンプレートを保存して、MapForce に挿入します。



次に留意してください:

現在の フラグメント内のテキストフラグメントが条件を満たす場合、フラグメントの**データ全体**が、関連したコンテナにパスされます。データは分割されず、関連したコンテナ、または、定義された条件を1つも満たさない場合、デフォルトのコンテナにルートされます。

「切り替え」により作成された関連したコンテナは、更に処理するために使用することができます。このようなコンテナを「1回分割する」、「繰り返される分割」、または、その他に分割することができます。

コンテンツの先頭:

テキストフラグメントの最初に条件文字列が表示される場合、関連したコンテナのみにデータがパスされます。

コンテンツに含まれる内容:

テキストフラグメントに条件文字列が表示される場合、関連したコンテナのみにデータがパスされます。

最初のトリガーされた条件のため:
条件リスト内の**条件**が**1つ**でも**true**の場合、trueである他の条件は、無視され、データは関連したコンテナにパスされません。

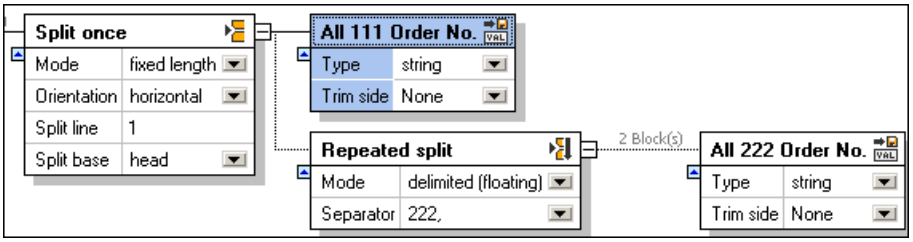
すべてのトリガーされた条件のため:
条件リスト内でtrueの**すべての**条件のためにデータを出力します。複数の関連したコンテナ内の同じデータフラグメントが同時に発生することを許可します。これは、テキストフラグメントが、2つの条件を同時に満たすテキストを含む場合、発生する可能性があります。例 上のサンプル内の『シーケンスの開始』。

7.4.5.4 ノード

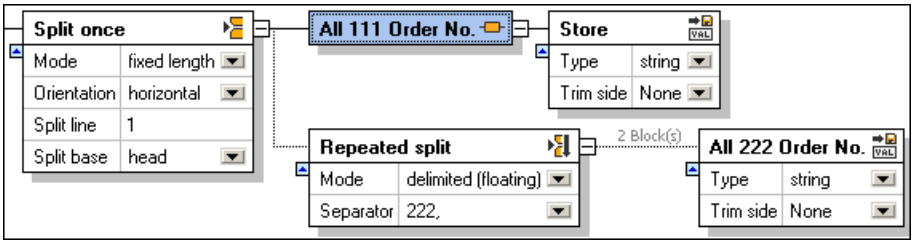


FlexTextとMapForceツリー構造に新しい階層構造レベルを追加することを許可します。次のノードにコンテナが含むデータは、以下のようにパスされます。

下のスクリーンショットでは、「All 111 Order No.」コンテナがトップのブランチの最後のコンテナです。



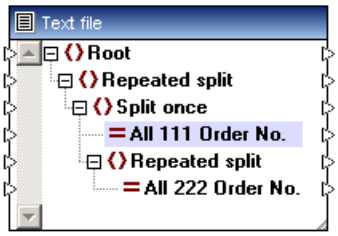
コンテナの右上をクリックして、コンテキストメニューから『**Node**』を選択します。

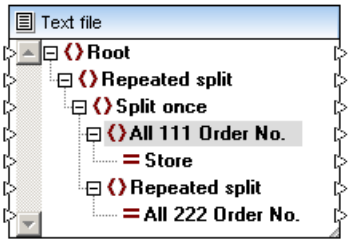


新しいコンテナが現在のコンテナの右に追加されました。

次に留意してください:
自動的に追加されたコンテナを、手動で**『値として保管する』**と定義することができます。

下のスクリーンショットは、MapForce内に挿入された時のテンプレート構造を表示しています。



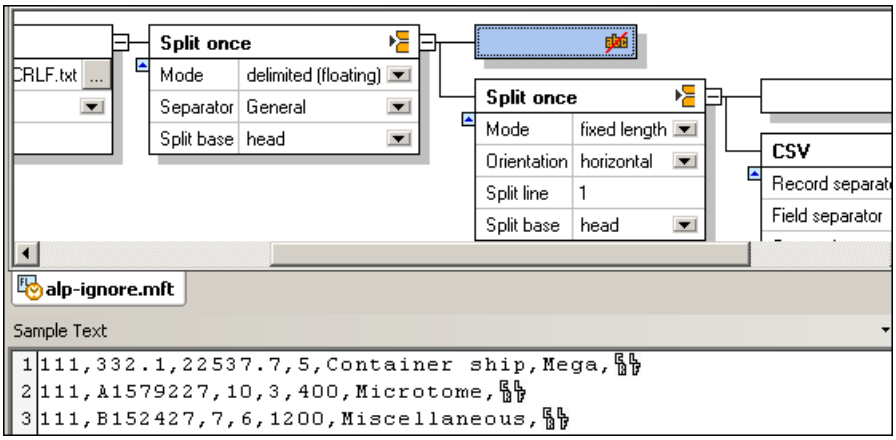


左のコンポーネントでは、新しいノードが追加される前の最初の構造が表示されています。

右側のコンポーネントは、コンポーネント構造が変更されていることを表示しています。「All 111...」が親アイテムになり、新しい子アイテム「保管」が追加されました。

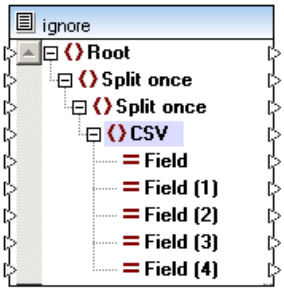
7.4.5.5 無視

特定のテキストフラグメントの出力を抑制することができます。データが含まれているコンテナがMapForce 内のFlexText コンポーネントで、マップすることのできるアイテムとして使用できないようにすることを意味します。

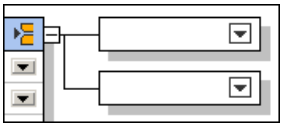


上に示される例では、アクティブなコンテナは、「無視する」に設定されています。このため、含まれるサンプルテキストは、MapForce 内のマップすることのできるアイテムとして表示されません

MapForce に挿入されたテキストテンプレートは、構造は下に表示されています。2つの「1回分割する」アイテムの間には、マップすることのできるアイテムは存在しません。



次に留意してください:
デフォルトの「無視する」コンテナも存在します。これらは、「1回分割する」と「繰り返される分割」などを選択すると、自動的に追加される新規のコンテナです。



これらのコンテナのコンテンツは、テンプレートが挿入された時点では、MapForce で使用することはできません。マップするためには、FlexText 内で次のコンテナオプションの1つを選択する必要があります :値として保管、CSV として保管 など。

7.4.5.6 CSV として保管 (区切られた)

CSV として保存することにより、テキスト フラグメントをCSV テキストとして保管し、MapForce で個別の列にマップすることができます。FlexText 内で作成することのできるCSV コンテナ/コンポーネントの数量には制限がありません。CSV

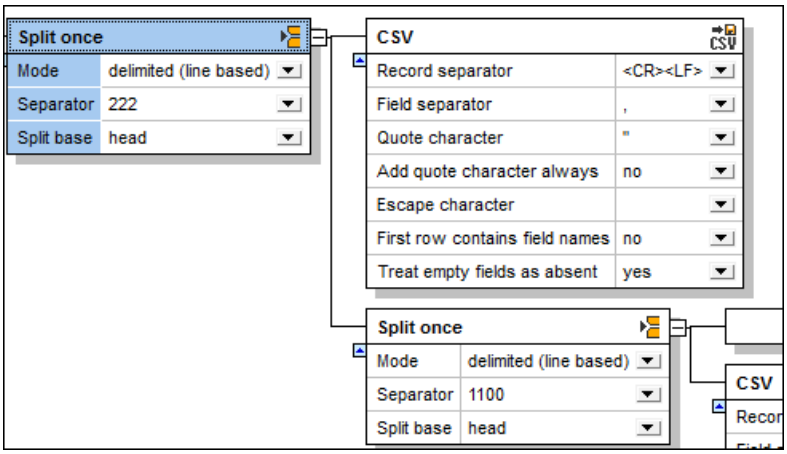
コンテナには、異なるセパレーターが存在する場合があります。

サンプルテキストペインは、現在のCSV フラグメントの概要を提供し、個別の フィールド名とフィールド型を指定することを許可します。 MapForce 内のFlexText コンポーネント内のマップすることのできるアイテムとして、各列は表示されます。

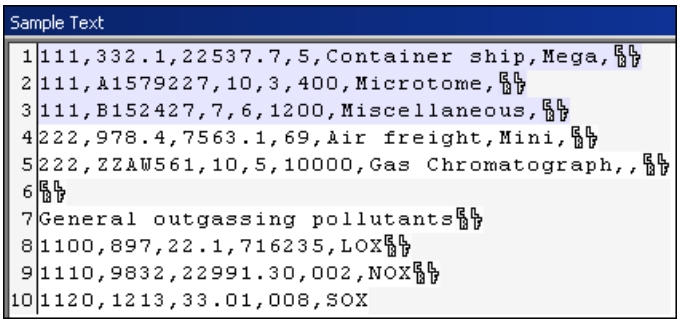
コンテナのデフォルトの設定：

レコードセパレーター	CR LF
フィールドセパレーター	,
引用 文字	"
引用文字を常に追加する	いいえ
エスケープ文字	(無し)
最初の行を含むフィールド名	いいえ
空のフィールドを不在として扱う	はい

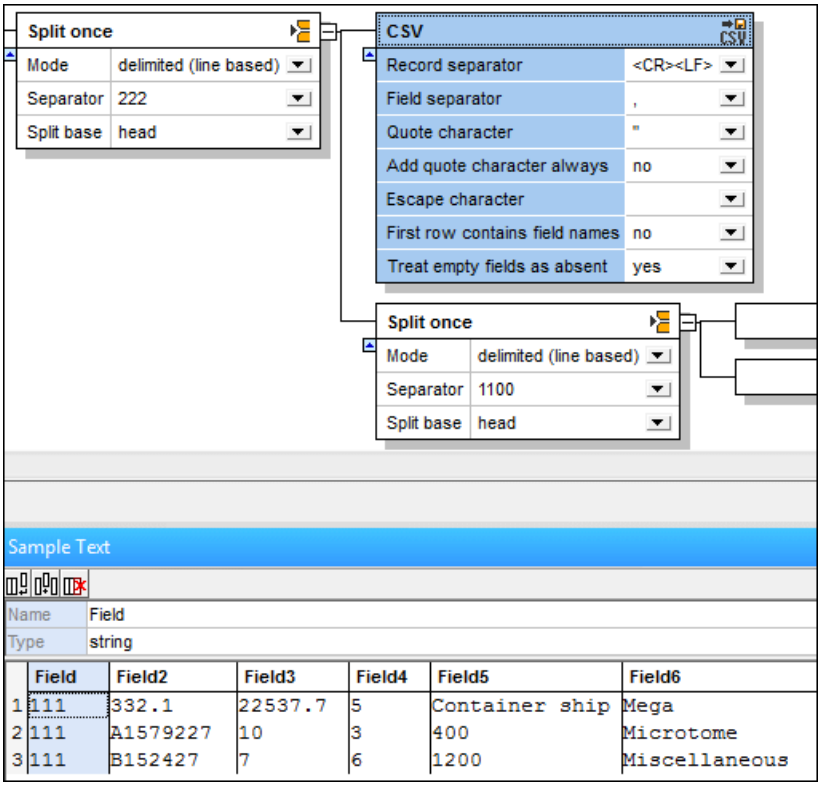
次のサンプルは、小さなテキストファイル内のデータを 2つのCSV ファイルに分割し、MapForce 内の個別のXML ファイルにマップする方法を示しています。



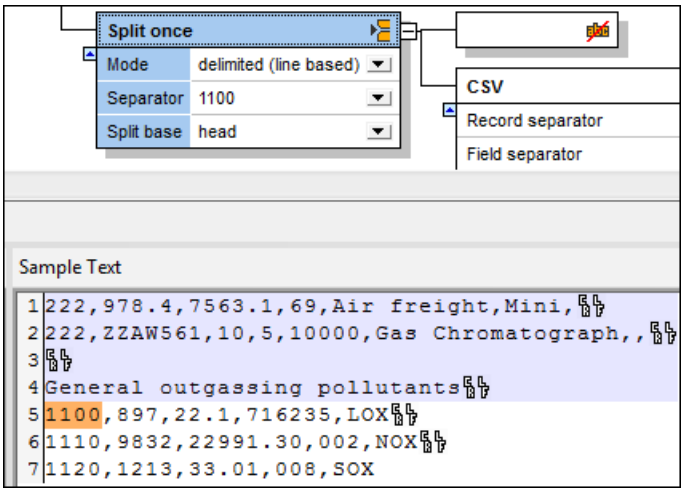
上に表示される、**「1回分割する」**コンテナは、2つのコンテナを作成するために使用されています。セパレーター 222 を持つ **区切られた (行ベース)**関数は、これを達成するために使用されています。222 の最初の発生までのすべてのレコードは、CSV コンテナにパスされます。 111 を含む すべてのレコードから構成されている最初のコンテナは、CSV コンテナとして定義されています。サンプルテキストペインは、現在 アクティブなコンテナ 「1回分割する」のコンテンツを表示します。



デフォルトのCSV 設定 は変更されていません。CSV コンテナクリックすると、コンテンツがテーブルのフォームで表示されます。

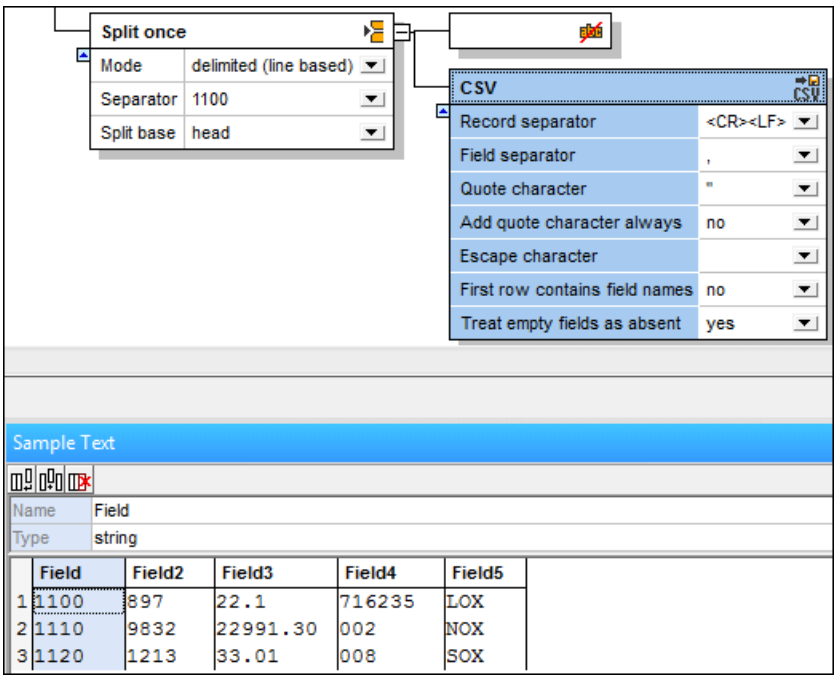


2番目の コンテナ には残りの データが保管されており、他の **「1回分割する」** コンテナが作成されます。これにより、2つのコンテナが作成され、そのうちの1つが 2番目のCSV になります。「1回分割する」 コンテナをクリックすると、現在の コンテンツ が表示されます。

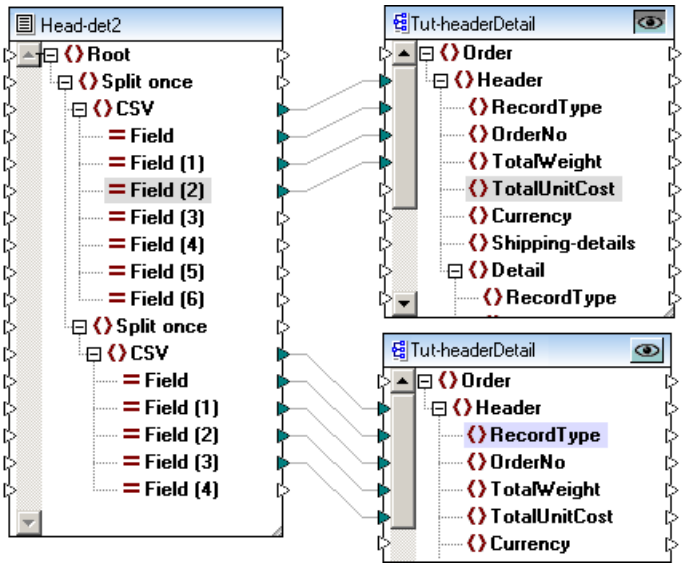


1100 をセパレーターとして使用して、区切られた (行ベース) 関数は、残りのデータを2つのセクションに分割するために使用されます。

- 1100 の最初の発生までのすべてのレコードは、 「無視する」 と定義することにより、マップすることのできない最初の コンテナ にパスされます。
- 2番目の コンテナはCSV として定義されています。デフォルトの設定 は変更されていません。CSV コンテナをクリックすると、コンテンツは、テーブル書式で表示されます。



MapForce にFlexText テンプレートを挿入すると、サポートされるターゲット ファイルにマップすることができます。このサンプルでは、CSV アイテムのそれぞれが、2つの個別のXML ファイルにマップされています。

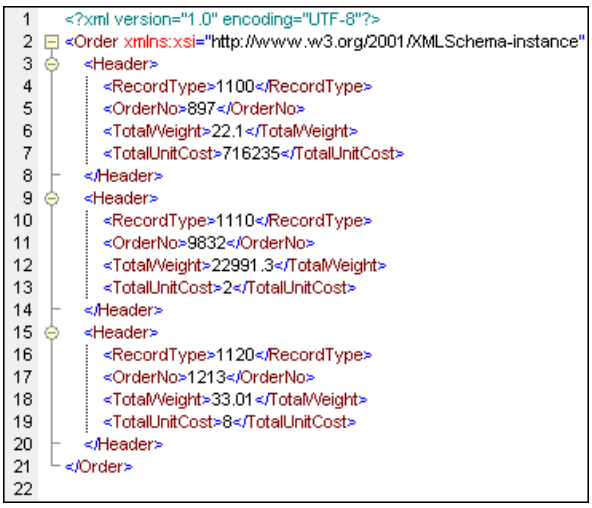


CSV セクション内のすべてのアイテムがターゲットファイルにマップされないことに注意してください。
最初のXML ファイルには、すべての 111 レコード型が含まれています。



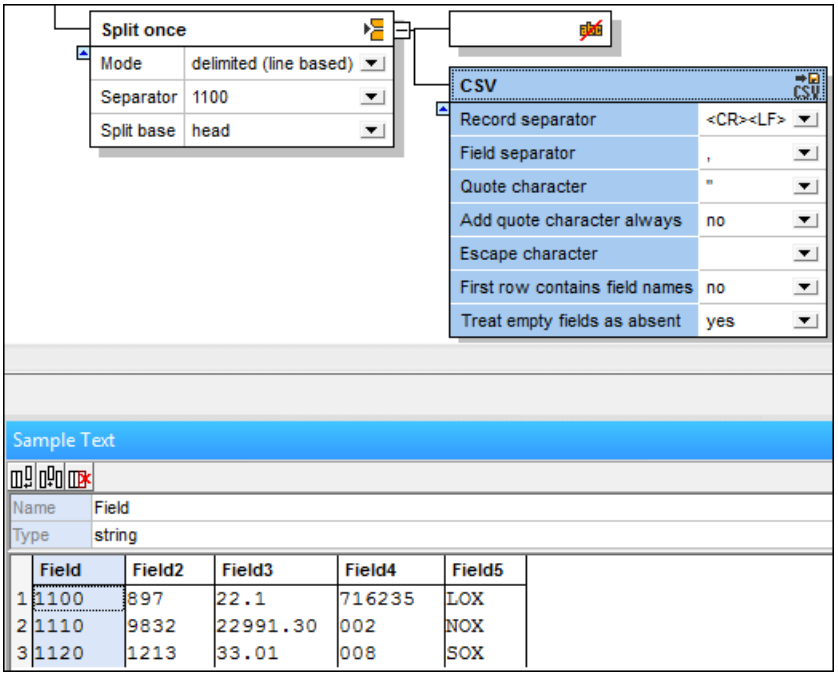
```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <RecordType>111</RecordType>
    <OrderNo>332</OrderNo>
    <TotalWeight>22537.7</TotalWeight>
  </Header>
  <Header>
    <RecordType>111</RecordType>
    <OrderNo>0</OrderNo>
    <TotalWeight>10</TotalWeight>
  </Header>
  <Header>
    <RecordType>111</RecordType>
    <OrderNo>0</OrderNo>
    <TotalWeight>7</TotalWeight>
  </Header>
</Order>
```

2番目のXML ファイルには、 1100から開始するすべてのレコードが含まれています。



```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <RecordType>1100</RecordType>
    <OrderNo>897</OrderNo>
    <TotalWeight>22.1</TotalWeight>
    <TotalUnitCost>716235</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1110</RecordType>
    <OrderNo>9832</OrderNo>
    <TotalWeight>22991.3</TotalWeight>
    <TotalUnitCost>2</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1120</RecordType>
    <OrderNo>1213</OrderNo>
    <TotalWeight>33.01</TotalWeight>
    <TotalUnitCost>8</TotalUnitCost>
  </Header>
</Order>
```

CSV コンテナデータの構成：



サンプルテキストペイン内のフィールドをクリックすると、ハイライトされ、更に構成することができます。

- **名前** フィールドをクリックして、存在するデフォルトのテキストを編集します。
- **型** フィールドをクリックして、次の フィールド データ型を定義します：文字列、ブール、小数など。
- 追加アイコン をクリックして、新規フィールドを追加します。
- 挿入アイコン をクリックして、現在 アクティブなフィールドの前にフィールドを挿入します。
- 削除アイコン をクリックして、現在アクティブなフィールドを削除します。

次に留意してください：
フィールド境界をマウスでドラッグして、データを表示することができます。

引用文字を常に追加する

指定された引用 文字が生成されたCSV ファイルのすべてのフィールドに追加するかを定義することを許可します。

空のフィールドを不在として扱う

ソースファイル内の空のフィールドを定義し、ターゲットファイル内に対応する空のアイテム（要素、または、属性）を作成することを許可します。

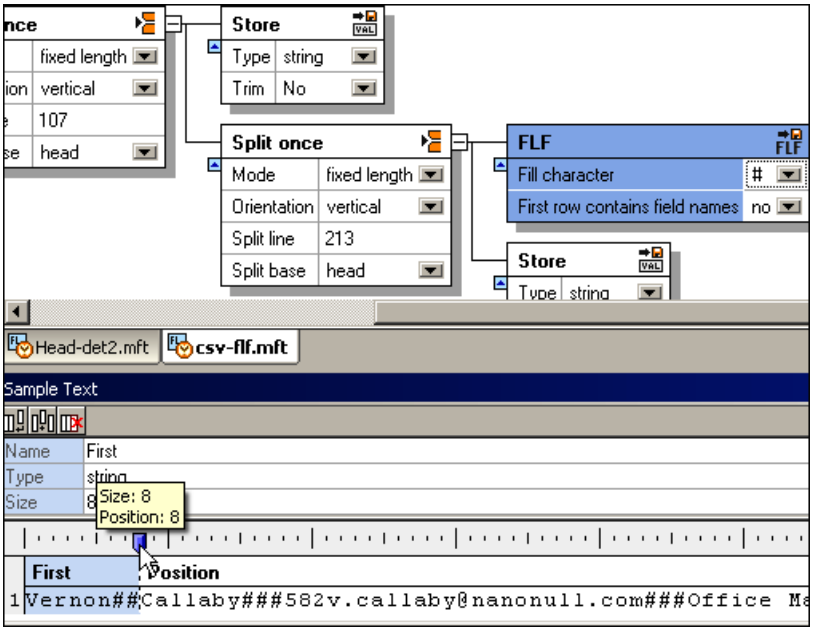
ソースファイル内の空のフィールドのための区切り文字は、存在する必要があります。例：「General outgassing pollutants,,,,」。

7.4.5.7 FLF として保管（固定長）

Store FLF は、テキスト フラグメントは、固定長 テキストとしてテキストフラグメントを保管し、MapForce の個別の列としてマップすることを許可します。FlexText 内で作成することのできるFLF コンテナ /コンポーネントの数量に制限はありません。各 FLF コンテナには異なるファイル文字が存在する場合があります。

サンプルテキストペインは、現在の FLF フラグメントの概要を提供し、個別の フィールド名、長さ 幅を指定することを許可します。 MapForce 内の FlexText コンポーネント内のマップすることのできるアイテムとして、各列は表示されます。

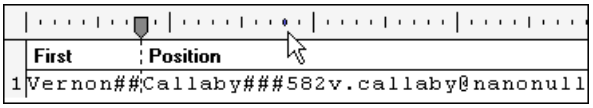
コンテナのデフォルトの設定：
フィル文字 (無し)
最初の行を含むフィールド名 いいえ
空のフィールドを不在として扱うはい



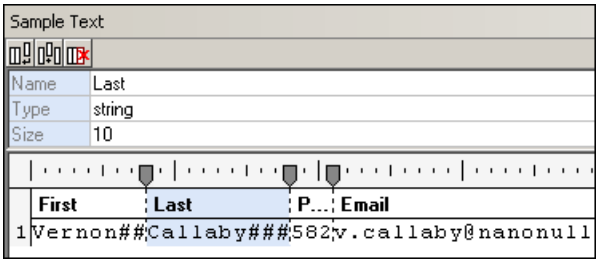
FLF コンテナ/データの構成

「FLF として保管」としてコンテナを定義すると、上のスクリーンショット内のテキストが、サンプルテキストペインに表示されます。 デフォルトの幅が 10 のフィールドが自動的に挿入されます。

- ルーラー上のタブ アイコンをクリックして、位置を移動します。現在の位置を示すヒントが表示されます。
- カーソルの位置をルーラー上で選択すると、「ドット」で位置が示されます。新しいタブの位置をクリックして位置を設定します。クリックします。

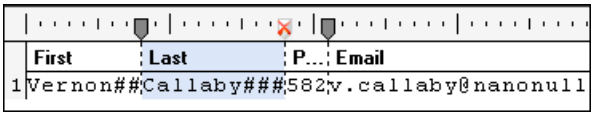


- 新しい位置を定義し、フィールドをクリックして選択し、名前 フィールド内の名前を編集します。



- フィールドを**削除**するには、タブ アイコンをクリックして、ルーラーからドラッグします。タブ アイコンは、このアクションの操

作に成功すると変更されます。



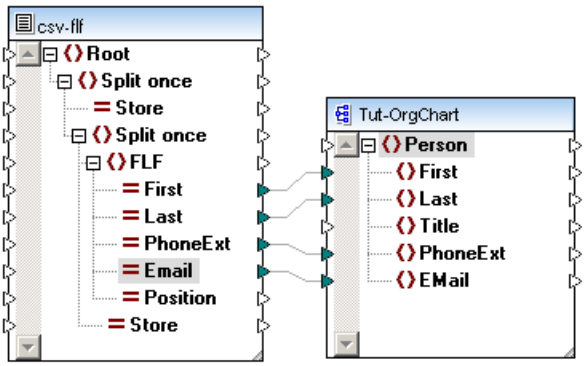
メモ: タブをルーラーにドラッグすると、右側のすべてのタブが自動的に調整されます。他のタブの位置を保持するには、**Shift** キーを移動キーを押す前に押します。

サンプルテキストペイン内のフィールドをクリックすると、ハイライトされ、更に構成を設定することができます。

- 追加アイコン をクリックして、新しい長さの 10 フィールドを追加します。
- 挿入アイコン をクリックして、現在 アクティブな、長さ 10 のフィールドの前にフィールドを挿入します。
- 削除アイコン をクリックして、現在 アクティブなフィールドを削除します。

- **名前** フィールドをクリックして、表示されるデフォルトのテキストを編集します。
- **型** フィールドをクリックして次のフィールド データ型を定義します :文字列、ブール、小数、など。

MapForce に FlexText テンプレートを挿入すると、サポートされるターゲット ファイルにデータをマップすることができます。このサンプルでは、FLF アイテムは、XML アイテムにマップされています。



オプション**空のフィールドを不在として扱う**が **「はい」** に設定されている場合、ソースファイル内の空のフィールドは、ターゲットファイル内に対応する空のアイテム、要素、または、属性)を作成しません。 2つのフィル文字の間にデータが存在しない場合、フィールドは不在として扱われます。

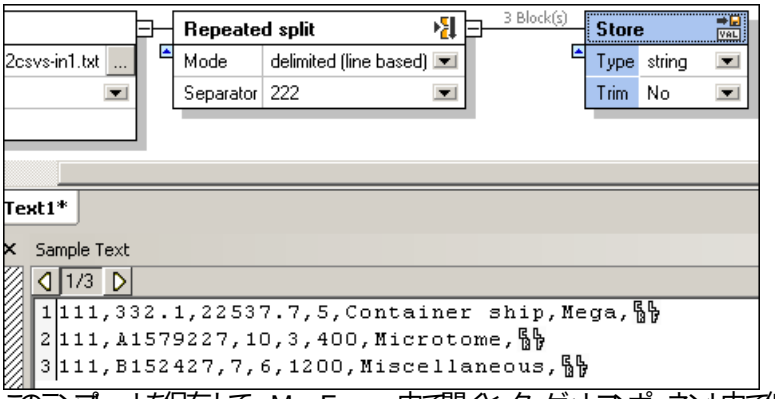
7.4.5.8 値の保管

MapForce 内でマップ可能なアイテムとしてデータを処理するコンテナを定義することができます。FlexText 内でコンテナの名前を変更しない場合、マップ可能なアイテムは 保管」という名前で表示されます。

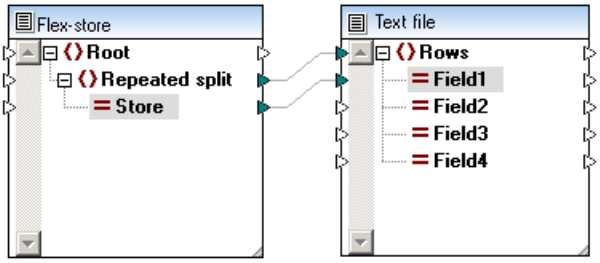
コンテナのデフォルトの設定 :

型	string
トリム	いいえ

下のスクリーンショットは、サンプルテキストペイン内で表示されているコンテンツがある 保管」コンテナを示しています。



このテンプレートを保存して、MapForce 内で開くと、ターゲットコンポーネント内で他のアイテムに保存されたアイテムをマップすることができます。



次に留意してください：

ターゲット テキスト ファイル内のフィールド1 アイテムは、結果をプレビューするために出力タブをクリックすると、保管アイテムにより与えられる 3 つ全てのフラグメントを含みます。

型
テキストフラグメントのデータ型を定義することができます。

サイドのトリム
文字がトリムされるサイドを定義します。左、右 または両方。「はい」を選択すると、「文字 セットのトリム」オプションを有効化します。

文字セットのトリム
このテキストフラグメントからトリムする文字を定義します。入力することのできる文字数に制限は無く、フィールドをダブルクリックして行います。入力された文字がフラグメントのトリムするサイドから削除されます。

7.4.6 FlexText と正規表現

MapForce FlexText 内で正規表現を以下のように使用することができます：

1. (一致するテキストがセパレーターとして役割を果たすことにより)一致が検索されるとテキストコンテナを分割する。一致が検索されると、テキスト コンテナを分割します。この場合、正規表現は、明示的にアンカーされます。このため、キャレット (^) とドル記号 ($) 文字は、使用されません。例えば、3つの連続した数字に一致する場合、`^[0-9]{3}$` の代わりに `[0-9]{3}` を使用します。
2. テキストに正規表現の一致が含まれる場合、「切り替え」コンテナからテキストをリダイレクトします。

次のMapForce ターゲット言語のFlexText コンポーネント内で正規表現を使用することができます：

- Built-in (マッピングをプレビューする際に使用されます)
- Built-in (MapForce Server 実行可能ファイルを実行する際に使用されます)
- コード生成言語 (C++、C#、Java)。これらの言語では、正規表現の高度な機能は、特定の言語の正規表現の実装により異なることに注意してください。

FlexText 内の正規表現 構文とセマンティクスは http://www.w3.org/TR/xmlschema-2/#regexs をベースにしており、MapForce コア関数 **tokenize-regexp** に類似しています。以下に注意してください：

- 分割 条件が連続する2つのシーケンスに一致する場合、FlexText はからの結果を間に作成します（この振る舞いは、正規表現を使用しない場合でも同じ振る舞いが取られます）。
- 正規表現 フラグ (http://www.w3.org/TR/xquery-operators/#flags) はサポートされません。

このセクションには以下のトピックが含まれています：

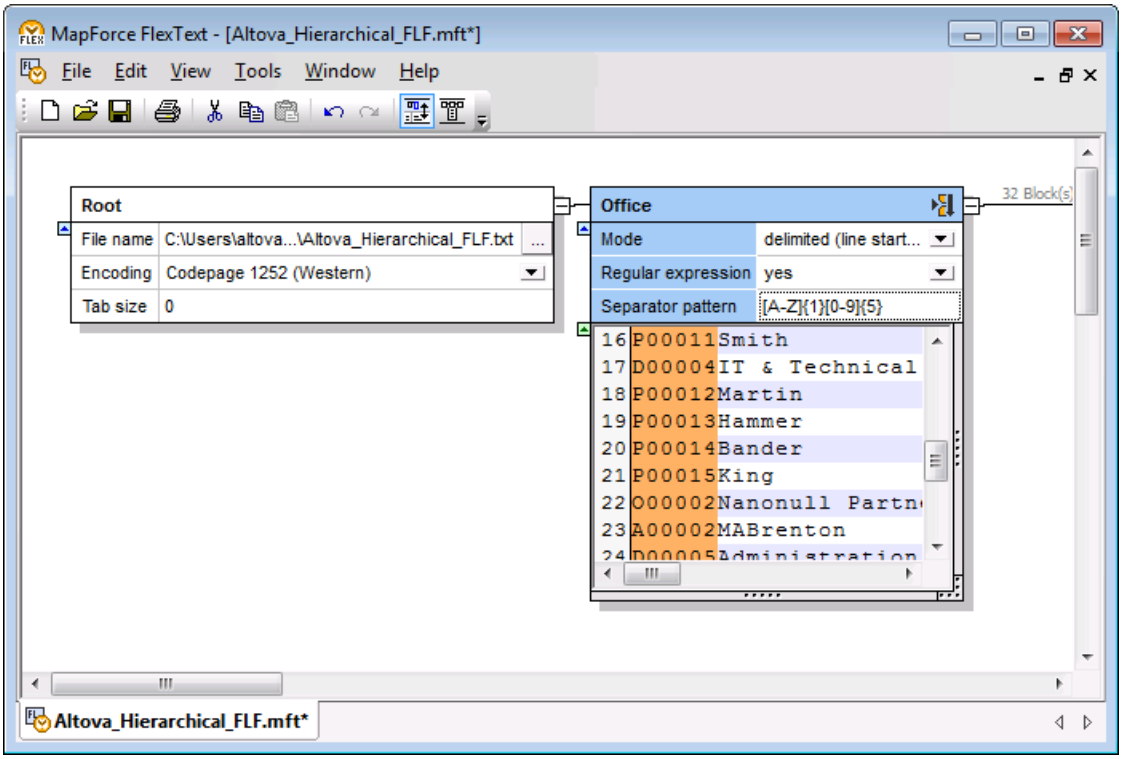
- 正規表現を使用してテキストを分割する
- 条件を変更するために正規表現を使用する

7.4.6.1 正規表現を使用してテキストを分割する

テキストを 2つの、または、複数のフラグメントに分割する場合、正規表現を任意でセパレーターとして使用することができます。これは、単一または複数の連続した文字によるテキストの分割などの更に高度なオプションの代替です。

次の条件が true の場合、正規表現を使用してテキストを分割するオプションは、FlexText 内で使用することができるようになります：

- コンテナは、型 **「1回分割する」** または **繰り返される分割** です。
- モード オプションが **区切られた（フロート）**、**区切られた（行ベース）** または、**区切られた（行の開始）** に設定されている場合。



個別のテキストに対して正規表現を使用するサンプルのFlexText テンプレート

正規表現に適用することのできるオプションは以下の通りです。

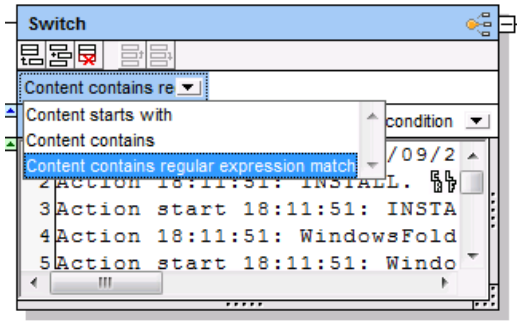
正規表現	テキスト セパレーターとして正規表現を使用するには、このオプションを **[はい]** に切り替えます。セパレーターとしてシンプルなテキストを使用する場合、このオプションを **[いいえ]** に切り替えてください (これはデフォルトの値ではありません)。
セパレーター パターン	*正規表現* オプションが **[はい]** に切り替えられていると、セパレーターとして振舞う正規表現を入力する箇所で、*セパレーター パターン* テキストボックスを使用できるようになります。上で説明されているFlexText テンプレートでは、正規表現 `[A-Z]{1}[0-9]{5}` が1つのアルファベット文字とその後に続く5つの数字文字 (例えば、「P00011」) が一致しています。この条件を満たすすべての発生は、プレビューでハイライトされ、テキストの分割セパレーターとしての役割を果たします。
書き込みのためのセパレーター	次の条件がtrue の場合、*書き込みのためのセパレーター* オプションには意味があります : • FlexText コンポーネントが ターゲット コンポーネントの場合 • *モード* が **区切られた (フロート)** に設定されている場合 正規表現のマッチが検出された箇所にあるターゲットコンポーネントに書き込まれる文字列をこのフィールドに入力します。

7.4.6.2 条件を変更するために正規表現を使用する

切り替えコンテナと作業する場合、 (次を参照 : 切り替え)、正規表現 の一致が含まれる場合、テキストフラグメントをリダイレクトするために、「切り替え」コンテナ内で条件を任意で作成することができます。 このような場合は、 C++ がターゲット変換言語として設定されている場合を除き、テキストの始めと終わりで一致が検索されるように、キャレット (^) とドル記号 ($) 文字を使用することができます。 (C++内で キャレットとドル記号 文字は、テキスト全体の始め、または、終わりではなく、行の始め、または、終わりとして解釈されます)。

条件を変更するために正規表現を使用する :

1. (コンテナの右上の角をクリックし、 **[切り替え]** をクリックして、) コンテナ 型を切り替えとして定義します
2. **[条件を追加する]** ( ) ボタンをクリックして、新しい条件を追加します。
3. 条件 型を **[コンテンツは正規表現 の一致を含む]** に設定します、隣接したテキストボックス内に正規表現を入力します。



サンプル
次に示されるデータベースログファイルからデータをマップすると仮定します (このファイルを次のパスで見つけることができます :

<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\SampleDatabaseLog.txt) 。

```
Action 18:11:51: INSTALL.
Action start 18:11:51: INSTALL.
Action 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Return value 1.
Action 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Return value 0.
Action 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Return value 0.
Action 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Return value 1.
Action 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Return value 1.
```

目標は以下のとおりです：

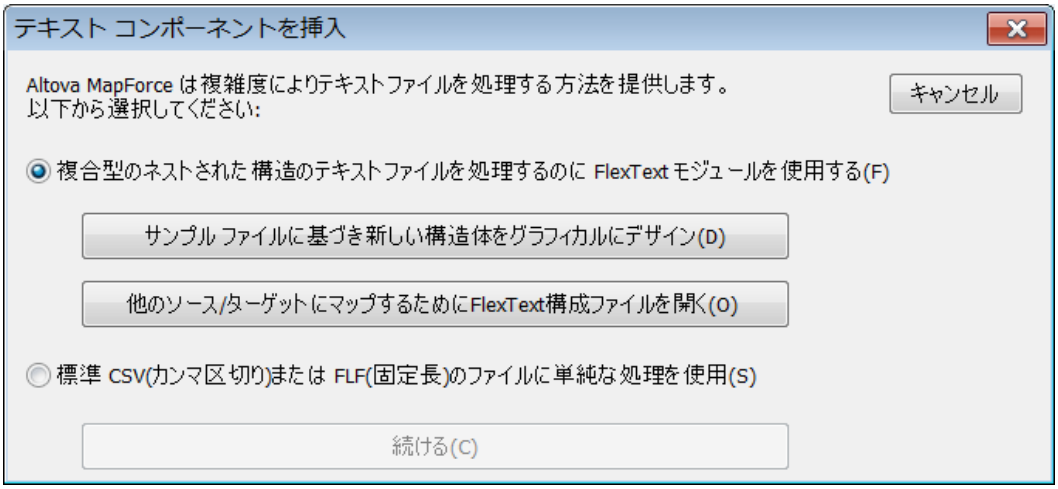
1. 戻り値が存在する箇所のすべての行を集めてリストにします。値「戻り値 1」または値「戻り値 0」、または、数値として表示される他の戻り値を含む行をリストは含みます。
2. テキストが値「Action start」を含む箇所のすべての行を他のリストに集めます。

この目的を達成するために、*繰り返される分割* コンテナを使用して、テキストを個別の行に分割します。この後、*切り替え* コンテナを使用して、各行を必要とされるようにリダイレクトします。具体的には、*切り替え* コンテナは、次に表示されるように 3つの条件から構成されています：

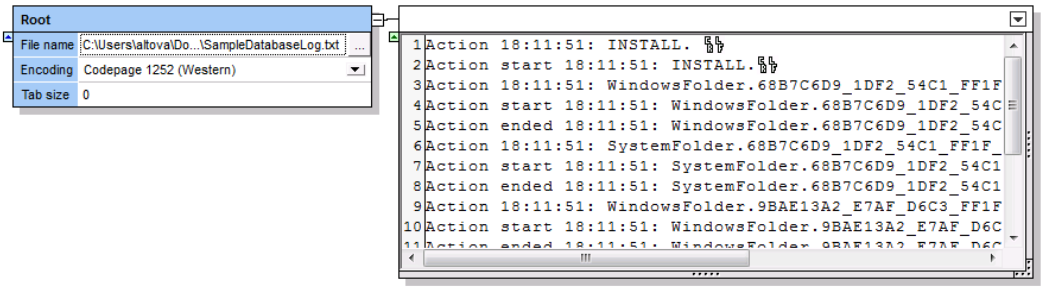
1. 出力 A が値「Action started」を含む場合、現在の行がリダイレクトされます。型「Content starts with」の条件を使用して、そのような行を検索し、「Action started」を値として入力します。
2. 出力 B が戻り値を含む場合、現在の行がリダイレクトされます。正規表現 `戻り値 [0-9]\.` を使用してこのような行を検索することができます。この正規表現は、行が単一の数字と終止符が後に続くテキスト「戻り値」を含む場合一致を返します。終止符がメタ文字ではなく通常の文字として扱われるように、終止符の前のバックスラッシュ (\) がエスケープ文字としての役割を果たします。
3. 行が上の条件を満たさない場合、現在の行を出力 C (<default>) にリダイレクトします。

上のタスクをおこなうFlexText テンプレートの作成：

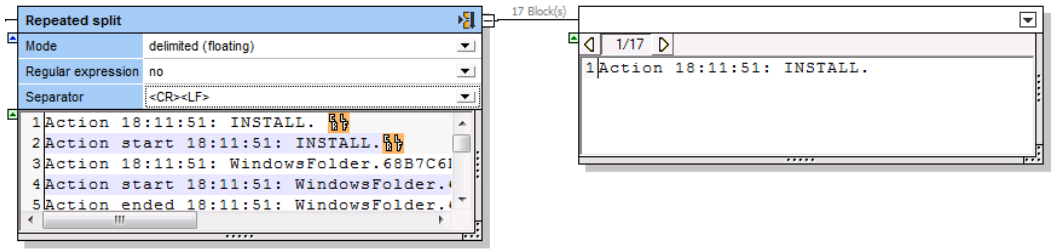
1. **「挿入」**メニューから、**「テキスト ファイル」**をクリックします。



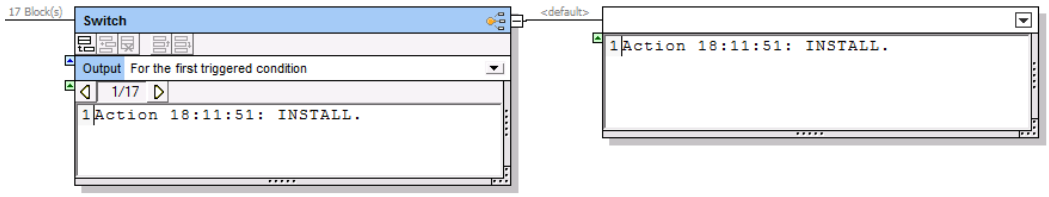
2. **『サンプルファイルに基づき新しい構造タイをグラフィカルにデザイン ..』**ボタンをクリックし、 FlexText .mft テンプレートを選択するディレクトリに保存します。
3. テキスト ファイルを開くようにプロンプトされると、**<Documents>\Altova\MapForce2017 \MapForceExamples\Tutorial\SampleDatabaseLog.txt** ファイルを参照してください。



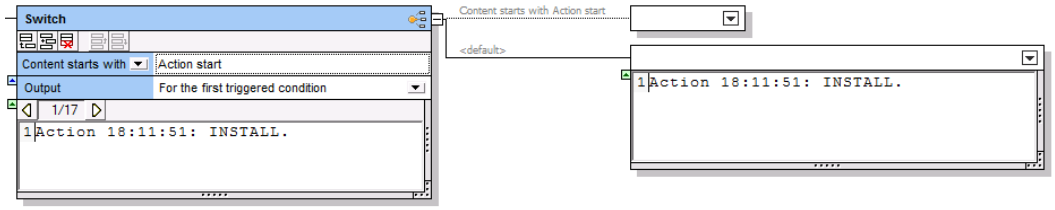
4. 出力コンテナの右上の角をクリックして、**繰り返される分割**を選択します。分割文字としてリターンキーを使用しているため、**区切られた (フロート)**モードを選択し、**<CR><LF>** をセパレーターとして選択します。 各行に1つの)17件のブロックにより構成されるこれにより新しい出力 コンテナが作成されます。



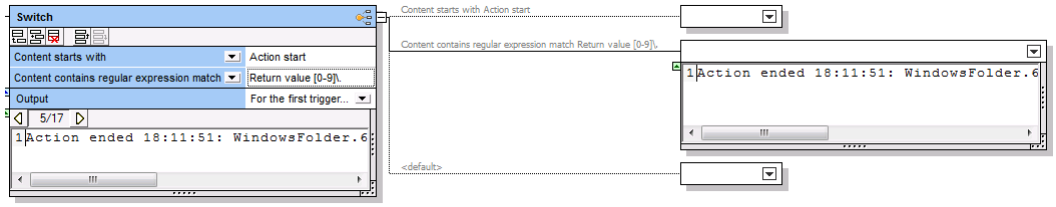
5. 新しい出力コンテナの右上の角をクリックして、**切り替え**を選択します。FlexText は、コンテナのコンテンツを置換えとして扱います。下に表示されるように、 1つの <default> 変更条件が自動的に作成されます。 この条件は、他の条件を満たさないのテキストを新しいコンテナにリダイレクトします。 この段階では、他の条件は定義されていません。 ですから、すべてのテキストは、現在 **<default>** 出力にリダイレクトされています。



6. **条件の追加**」( ) ボタンをクリックして、下に示されるように値 「Action start」を持つ型 「**Content starts with**」 条件 を追加します。 この条件 は、 「Action start」から開始するテキストを新しいコンテナにリダイレクトします。

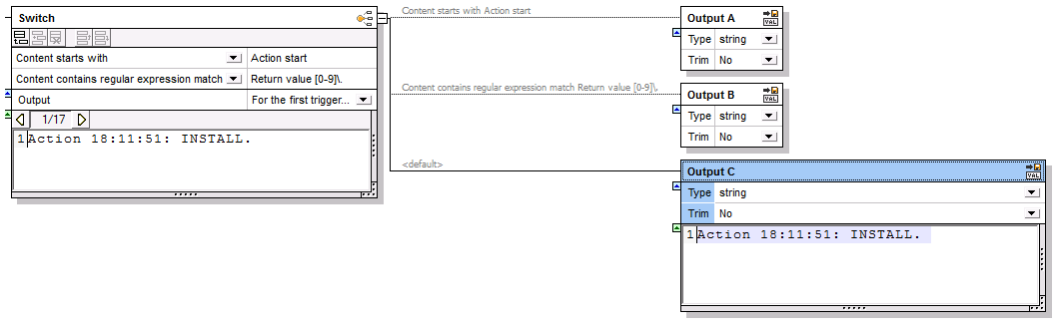


7. **条件の追加** ( ) ボタンをクリックして、 戻り値 `[0-9]\.` を持つ型 「**コンテンツは正規表現 の一致を含む**」の条件を追加します。 17件中 5番目のブロックへ移動する場合、ブロックには正規表現のための一致が含まれるため、この条件は、ブロックを新しいコンテナにリダイレクトされます。



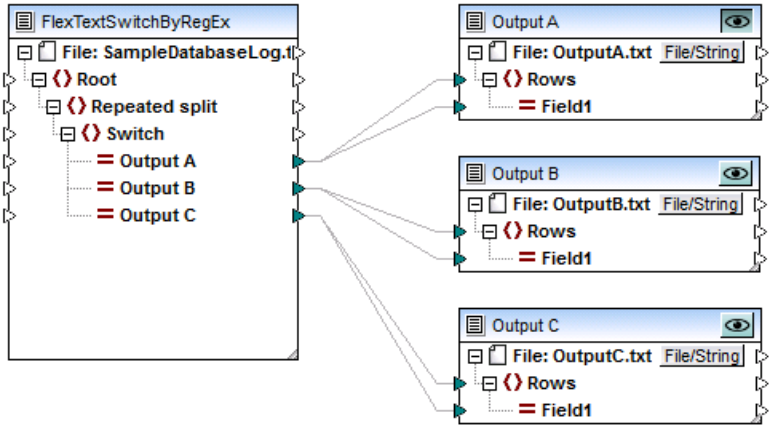
条件をベースにした異なる出力へテキストをリダイレクトするために、切り替え コンテナが構成されました。 3つの切り替え条件と3つの可能な 出力 (各条件に対してそれぞれ1つの出力 が存在します。 このチュートリアルの残りのステップは、個別のテキストファイルにそれぞれの出力を書き込む方法について説明しています。

8. 出力 コンテナの右上の角をクリックして、 **値として保管**」を選択します。更に、各出力のタイトルバーをダブルクリックし、名前を入力します :**出力 A**、**出力 B**、と**出力 C**。



9. FlexText テンプレートを保存して閉じます。

必要とされるFlexText テンプレートが作成されました。次のチュートリアルファイルを開き、作業することにより、どのようにこのテンプレートが作動するかを確認することができます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\FlexTextSwitchByRegEx.mfd**。



FlexTextSwitchByRegEx.mfd

スクリーンショット内で表示されている**FlexTextSwitchByRegEx.mfd** マッピングは、テキストデータを**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\SampleDatabaseLog.txt** ファイルから読み取り、3つの個別のテキストファイルに変換します：出力 A、出力 B、と出力 C。各ターゲット テキストファイルには、FlexText テンプレートで以前に定義された 3つの条件の1つを満たす行のみが含まれます。特別のターゲット コンポーネントの出力を確認するには、コンポーネントの右上の角の**コンポーネントのプレビュー**（ ）ボタンをクリックして、そして**出力** タブをクリックします。

7.5 EDI

Altova Web サイト EDI Translator

EDI (Electronic Data Interchange) は、機関とビジネスの間の電子データを有効化する一連の標準です。

MapForce は、以下の変換言語内でデータへの、またはEDI フォーマットデからの変換をサポートします :BUILT-IN、C#、C++、Java (変換言語の選択 も参照)。EDI データは、マップすることができ、データベース,フラットファイル, XML ドキュメントなど MapForce がサポートするフォーマットにマッピング、またこれらのフォーマットからマッピングすることができます。

MapForce は以下のEDI をサポートします :

標準	説明
ASC X12	ASC X12 はドキュメントのインターチェンジのための業界の標準です。MapForce は以下のバージョンをサポートします : 3040、3050、3060、3070、4010、4020、4030、4040、4041、4042、4050、4051、4052、4060、5010、5011、5012、5020、5030、5040、5050、6010、6020、6030、および 6040。 **MapForce のデフォルトの ASC X12 バージョンは 6040 です。** ASC X12 コンポーネントには、EDI 検証 設定ダイアログボックス内で選択された設定により EDI パーサーエラー情報やデータが書き込まれる「仮想」ノードがあります (EDI コンポーネントの検証を参照)。X12 997 Functional Acknowledgement はすべてのX12 ドキュメントから生成することができます。 ASC X12 に関しての詳しい情報は、http://www.x12.org/ を参照してください。
HIPAA X12	HIPAA はX12 EDI 5010 標準をベースにしていますが、MapForce 2011 リリース 3 または以降によりネイティブにサポートされている自身の特別なバージョンがあります。 **MapForce 内のデフォルトの HIPAA X12 バージョンは HIPAA 実装スペック (TR3) のリリース A2 です。** Altova Web サイトのMapForce コンポーネントページで前のリリースをダウンロードすることができます (http://www.altova.com/ja/components_mapforce.html)。
HL7	HL7 は医療アプリケーションのデータ交換のための業界の標準で、"Health Level Seven" の省略形です。MapForce はバージョン 2.2 から 2.6 をサポートします。 **MapForce 内のデフォルトの HL7 バージョンは 2.6 です。** 追加 HL7 V2.2 - V2.5.1 XML スキーマと構成ファイルのための個別のインストーラーが Altova Web サイト MapForce コンポーネントのページ (http://www.altova.com/ja/components_mapforce.html) で使用することができます。 XML-ベースのHL7 バージョン 3x は、XML スキーマ コンポーネントを使用して、MapForce 2017 内でサポートされています。 **MapForceExamples** プロジェクトは、HL7 V2.6 からHL7 V3 XML ファイルへマップするサンプル (HL7V260_to_HL7V3.mfd) が含まれています。 HL7 に関しての詳しい情報は、http://www.hl7.org/ を参照してください。

標準	説明
IATA PADIS	PADIS (Passenger and Airport Data Interchange Standard) は、EDI ドキュメントを使用した乗客と飛行場のデータのやり取りのための業界の標準です。 MapForce は、以下のバージョンをサポートします : 00.1 から 08.1。 **MapForce 内のデフォルトの PADIS バージョンは 08.1 です。** IATA PADIS に関しての詳しい情報は、 http://www.iata.org/Pages/default.aspx を参照してください。
SAP IDocs	SAP IDocs (中間ドキュメント) ドキュメントは SAP R/3 と非 SAP アプリケーション間でビジネスデータを交換する際に使用されます。ドキュメントは異なったシステム間で交換することのできる中間データストレージです。 SAP IDocs に関しての詳しい情報は、 http://help.sap.com/saphelp_nw70/helpdata/en/0b/2a6095507d11d18ee90000e8366fc2/frameset.htm を参照してください。
TRADACOMS	TRADACOMS (Trading Data Communications) は、小売業で使用される英国特有の電子データインターチェンジの標準です。 MapForce は、ベース TRADACOMS 仕様を Article Numbering Association (ANA) UK により出版された GS1 UK (https://www.gs1uk.org) として知られる 1993 年 1 月に出版された "電子データインターチェンジのための標準の TRADACOMS マニュアル" 内で指定されたとおり実装します。またはその他の TRADACOMS バージョンは、MapForce は、新規のメッセージの型、データ要素、コードの値、を構成ファイルによりカスタム化することができます。
UN/EDIFACT	UN/EDIFACT はドキュメントインターチェンジのための事実上の金融業界の標準 (また UN 標準) です。MapForce は、D93.A - D.16A ディレクトリ内に含まれるメッセージをサポートします。 **MapForce 内のデフォルトの UN/EDIFACT バージョンは D.16A です。** D93A までさかのぼる全てのバージョンの個別のインストーラーが Altova Web サイトの MapForce コンポーネントページで使用することができます (http://www.altova.com/ja/components_mapforce.html)。 UN/EDIFACT に関しての詳しい情報は、 http://www.unece.org/trade/untdid/welcome.htm を参照してください。

各 MapForce リリースは、(上のテーブル内で太字にハイライトされている) 各 EDI フレーバーのための通常最も新しい EDI 構成ファイルのバージョンが搭載されています。必要であれば、Altova Web サイトの MapForce コンポーネントページから EDI 構成ファイルの前のバージョンをインストールすることができます (http://www.altova.com/ja/components_mapforce.html)。インストールの後、インストールされた EDI バージョンはデフォルトのバージョンに加え MapForce で使用することができます。

マッピングエリアに追加した、単一の EDI コンポーネントのために、MapForce は、単一の標準 EDI リリースの複数のメッセージの型を処理することができます。例 : これにより 1 つ以上のメッセージ型を含むデータを交換可能なファイルから読み込むことができます。また、異なったメッセージの型を含む複数のインスタンスファイルからデータを処理することも可能です。上記のオペレーションは基になる仕様がサポートされる場合のみ意味を成します。

所属機関がEDI 仕様を使用する場合、カスタム化はすでにMapForce によりサポートされています。また、構成ファイルを使用して、カスタムEDI 仕様に対してMapForce が対応することもできます。例 :カスタムメッセージ、セグメント、データ要素、コードリストなどを追加（既存のものを変更）することができます。ですから、MapForce により既にサポートされているEDI に関連したディレクトリに類似したカスタムEDI コレクションを作成することができます。詳細に関しては、次を参照してください :EDI 構造のカスタム化 とEDI 検証のカスタム化。

7.5.1 EDI 用語解説

次のリストでは、EDI に関連した標準と共に使用される一般に使用される用語について説明されています。

メッセージ
メッセージとは、ビジネスドキュメントを表す集合のことです。ASC X12 内では、メッセージは、"トランザクション"と呼ばれます。メッセージは、"セグメント" と呼ばれる複数のビルディングブロックから構成されます。MapForce では複数種類のメッセージが単一のEDI コンポーネントにてサポートされます。（次を参照 :EDI コンポーネントをマッピングに追加する）。

セグメント
セグメントとは、メッセージに含まれる単一の「レコード」のことです。

セグメントは、2 または 3 文字からなるID により識別されます。関連する要素のグループによりセグメントタグ（ANSI X12 におけるセグメントID）が構成されます。トランザクションのセグメントは、必須または条件（オプション）セグメントとして定義することができます。

データ要素
セグメント内にある個々のデータフィールドのことです。要素はフィールドとして考えることができ、例えば 1 種類のデータ 名前、または住所などを含めることができます。要素は更に、コンポーネントの要素やサブ要素からなる複合要素へ分けることができます。

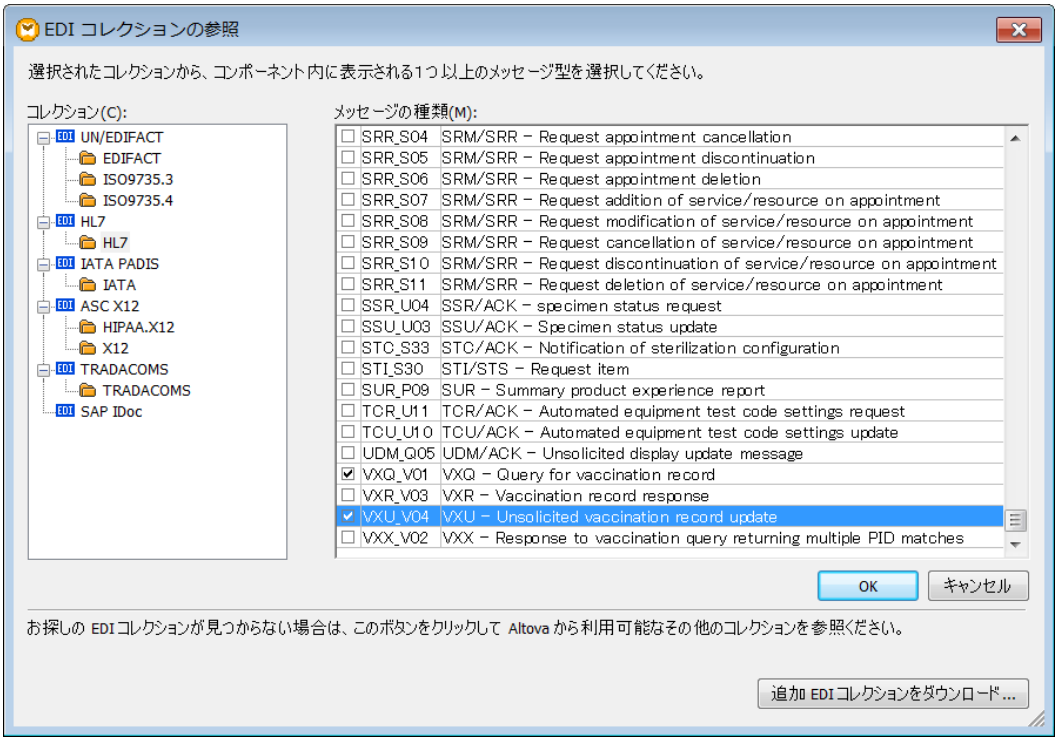
セパレーター
要素は「セパレーター文字」で区切られます。セパレーターは、通常テキストフィールドで発生するASCII 文字です（例えば、コロン、アスタリスク、パイプ バー、アンパサンドなど）。EDI フレーバーによりセパレーターは異なります。デフォルトでは、各 EDI フレーバーのために適用することのできる標準のセパレーターをMapForce は認識します。しかしながら、セパレーターは必要に応じてカスタマイズすることができます（次を参照 :EDI コンポーネント設定）。

7.5.2 EDI コンポーネントをマッピングに追加する

EDI ファイルからデータを、または、EDI ファイルへデータをマップするには、EDI コンポーネントを最初にマッピングに追加する必要があります。EDI コンポーネントは、マッピング上の EDI メッセージの構造を表しています。EDI コンポーネントがマッピングに追加されると、MapForce によりサポートされるコンポーネント型と同様にマッピングの接続を追加することができます（次を参照 :コンポーネントと作業する）。必要であれば、複数のメッセージを含むために EDI コンポーネントを構成することができます。これにより、複数の入力または出力ファイルを動的に処理 で説明されているように、複数のEDI ファイル からデータ（各ファイルから1つのメッセージ）を読み取る、または、同じファイルから異なるメッセージを処理することができます。

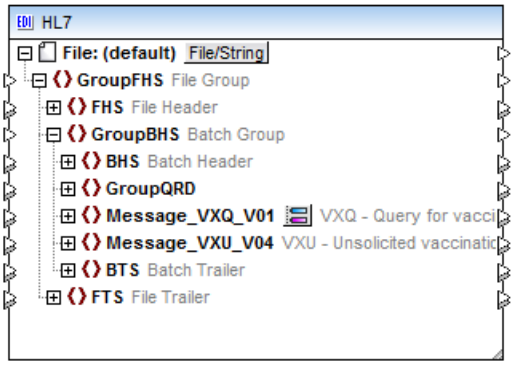
EDI コンポーネントをマッピングに追加する :

1. **「挿入」**メニューから、**「EDI」**をクリックします。
1. EDI コレクションをクリックし、含むメッセージの型の横のチェックボックスをチェックします。複数のメッセージを、対応するチェックボックスを選択して行います。例えば、下のイメージ内では、HL7 コレクションの 2件のメッセージがマッピングに追加されています。



2. 「OK」をクリックします。サンプル EDI ファイルを提供するように問われます。
3. EDI コンポーネントからデータを読み込む場合、サンプル EDI ファイルを参照します。それ以外の場合、 **「スキップ」**をクリックします。

選択済みのメッセージの構造を含むコンポーネントがマッピングに表示されます。追加メッセージを追加する場合、または、メッセージを削除する場合、最初のメッセージの横に表示される ボタンをクリックします。



2 つのメッセージを含むサンプル HL7 マッピングコンポーネント

この EDI コンポーネントからマッピング接続を描くことができます。複数の EDI フレーバーからのマッピングを表す部の MapForce マッピングデザイン (.mfd) ファイルは、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内で確認することができます。

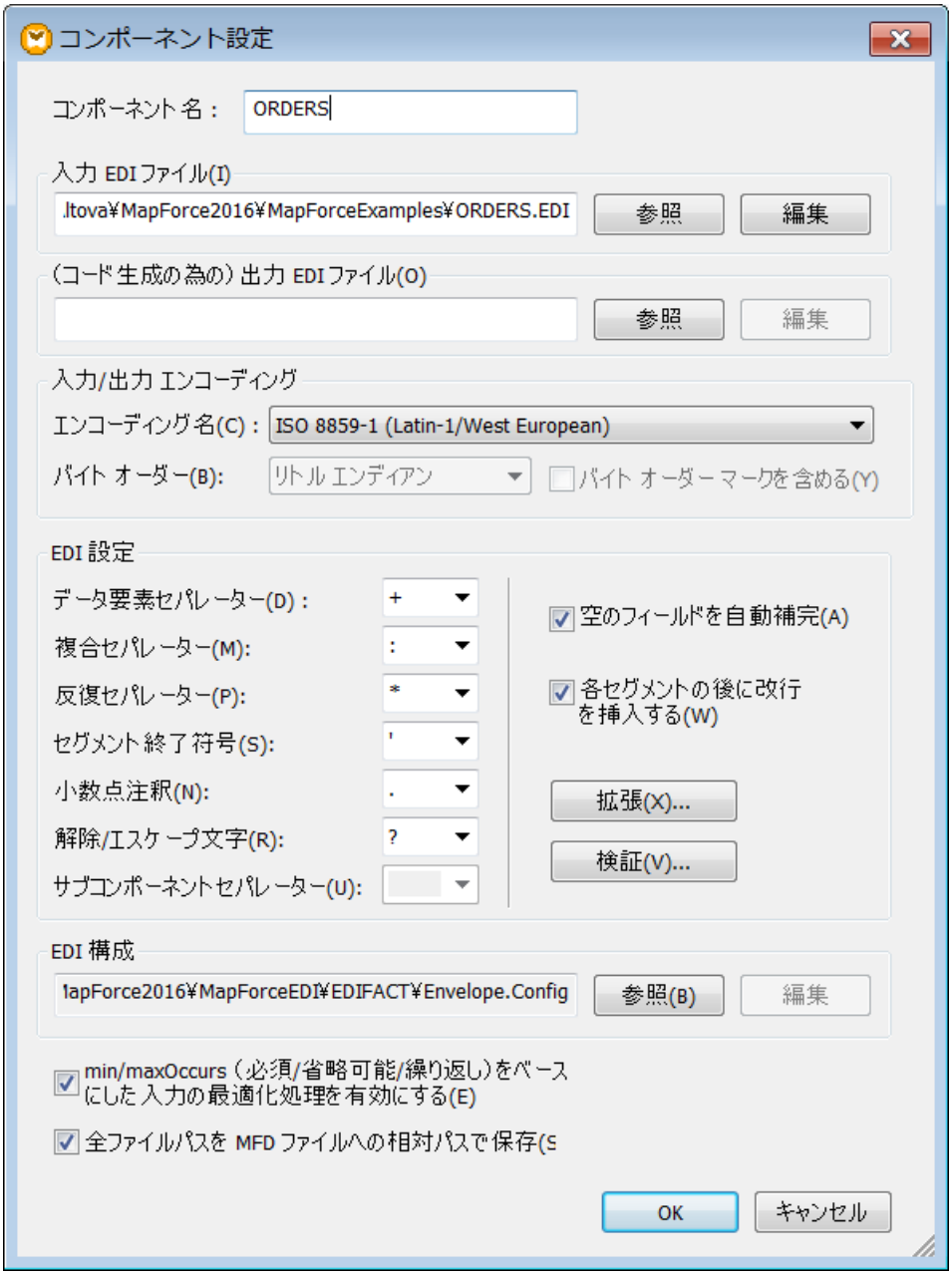
EDI フレーバー	マッピングサンプル
ASC X12	• **X12_To_Xml_Order.mfd** • **Xml_To_X12_Order.mfd**

EDI フレーバー	マッピングサンプル
HIPAA X12	• **HIPAA_837D.mfd**
HL7	• **HL7_MultiMessageTypes.mfd** • **HL7V260_To_HL7V3.mfd**
IATA	• **IATA_FlightInformationReport.mfd**
SAP IDoc	• **IDOC_Order.mfd**
UN/EDIFACT	• **EDI_Order.mfd** • **EDI_Order_DB.mfd** • **EDI_OrderReverse.mfd**

7.5.3 EDI コンポーネント設定

EDI コンポーネントをマッピングエリアに追加するには、コンポーネント設定ダイアログボックスから適用することの出来る設定を構成します。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます：

- マッピング上のコンポーネントを選択し、 **「コンポーネント」** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。



EDI コンポーネント設定 ダイアログボックス

使用することのできる設定は以下にリストされています。現在選択されているコンポーネントのEDI 特徴がサポートされない場合、一部の設定を使用することはできないことに注意してください。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名が自動的に生成され、インスタンスファイルをその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。

	コンポーネント名はスペースを含むことができ（例："ソースXML ファイル"）またフルストップ文字も含むことができます。（例："Orders.EDI"）。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、先頭および行末のスペースを含むことはできません。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください： • マッピングをFlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとWindows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
入力 EDI ファイル	MapForce がデータを読み込むEDI ソースファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、EDI インスタンスファイルに割り当てます。
出力 EDI ファイル	MapForce がデータを書き込むEDI ターゲットファイルを指定します。このフィールドはターゲットコンポーネントにとって重要です。
入力 /出力エンコード	出力インスタンスファイルの次の設定を指定することを許可します： • エンコード名 • バイトオーダー • バイトオーダーマーク（BOM）文字が含まれるか否か。 デフォルトでは、**新しいコンポーネントのためのデフォルトのエンコード**」オプション内で定義されたエンコードを持つ新しいコンポーネント。**「ツール	オプション」**からこのオプションにアクセスし、タブを生成します。
EDI 設定	この設定グループにより、カスタム区切り文字、セパレーター、終端文字を定義することができます（EDI フォーマットによりサポートされている場合のみ、この設定を使用することができます）。
不足するフィールドの自動補完	このオプションは ターゲットEDI コンポーネントに適用されます。このチェックボックスが選択されると、MapForce は、データフィールドの一部を自動的に補完します。これは、操作が現在選択されているEDI フォーマットの仕様に相反しない場合のみ適用することができます。このような振る舞いを無効化するには、このチェックボックスをクリアしてください。
各セグメントの後の改行	このオプションは ターゲットEDI コンポーネントに適用されます。チェックボックスが選択されると、MapForce は、CR/LF（リターンキー /開業）文字を、各 EDI セグメントの後に追加します。
追加設定	現在の現在のEDI コンポーネントの追加設定を定義することのできるダイアログボックスを開きます 使用することのできる設定はEDI 種類により異なります。
検証	現在のEDI コンポーネントの検証設定を定義することのできるダイアログボックスを開きます（EDI コンポーネントの検証を参照）。
min/maxOccurs をベースにした入力処理の最適化の有効化	このオプションは、minOccurs とmaxOccurs="1" などの必要な属性または子要素が1つのアイテムのみを含むシーケンスのための特別な処理を許可します。この場合、シーケンスの最初のアイテムが抽出されると、アイテムは（シーケンスとしてではなく）直接動的な値として処理されます 入力データがスキーマに対して**有効でない場合、**マッピングをエラーメッセージとともに停止する空のシーケンスがマッピング内で生じる場合があります。このような**無効な入力**を処理するには、このチェックボックスをクリアしてください。

MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定 は、以下のファイルに影響を与えます : • 入力 EDI ファイル (存在する場合) • 出力 EDI ファイル (存在する場合) 以下も参照 :コンポーネント上で相対パスを使用する。

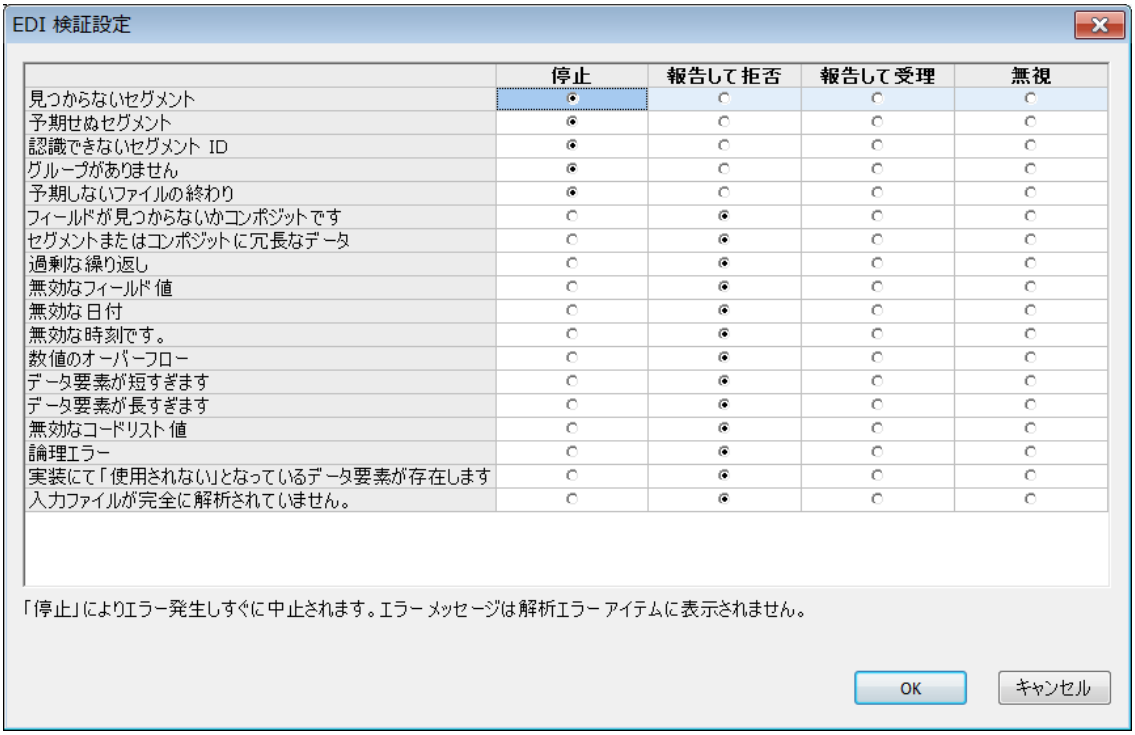
7.5.4 EDI コンポーネント検証

MapForce は、マッピングが実行される際にサポートされる全てのEDI ソースとターゲットドキュメントを検証します。これは MapForce 内でマッピングをプレビューする際に、出力タブをクリックして、または、MapForce Server、またはC++、C#、や Java プログラムなどの外部の実行エンジンにより実行される場合に発生します。マッピングにより生成されたEDI 出力のみを検証することもできます。これを行うには、 **出力」**タブをクリックして、 **出力の検証」** ツールバーボタンをクリックします (次も参照 : マッピング出力の検証)。

MapForce は、EDI ドキュメントをこのドキュメント内で指定されているとおり、個々のサポートされるフォーマットのために検証します。検証処理は、しかしながら、完全なEDI 構文またはセマンティックの検証ではありません。例えば、使用することのできるフィールドの値が、処理の可能性を予想する無限かつ完全な値のセットで簡単に示すことができない場合があります。このような場合、構成ファイルを使用して、ケースバイケースで検証ルールをカスタム化することが可能です (次を参照してください : EDI 検証のカスタム化)。

更に簡単なレベルでは、検証パラメーターは、MapForce から特定のニーズに対応するために直接カスタム化することができます。例えば、エラーを無視、特定のエラーでの検証処理の中止、エラーの受け入れ、拒否、または報告などを選択することができます。

検証設定をカスタム化するには、EDI コンポーネントをダブルクリックして、コンポーネント設定 ダイアログボックス内の **検証」** ボタンをクリックします。 (次も参照 : EDI コンポーネント設定)。デフォルトの検証設定 は、下に示されています。



EDI 検証設定 ダイアログボックス

上にリストされる検証イベントでは、以下のアクションをとることができます :**停止、報告して拒否、報告して受理、無視**。

停止 : 致命的なエラーをキャッチして、レポートメッセージを生成すること無くマッピングの実行を停止します

報告して拒否 は、エラーの情報を提供し、現在のメッセージを拒否します。**報告して受理** は、エラーの情報を提供し、現在のメッセージを受理します。実際のエラーは、MapForce のメッセージウィンドウ内で確認することができます。ソース X12 とHIPAA コンポーネントに関しては、検証エラーは、X12 997 または X12 999 レポートを作成するマッピングコンポーネント内でログされます (次を参照 : 検証結果とX12 Acknowledgement)。

MapForce Server または、生成されたプログラムコードでマッピングが実行される場合、エラーを解析するとマッピングを停止する例外がスローされ、コマンドラインに検証エラーが表示されます。

無視 は、検証イベントを無視します。このアクションを選択すると、エラーに関しての情報は表示されませんので、注意して使用してください。

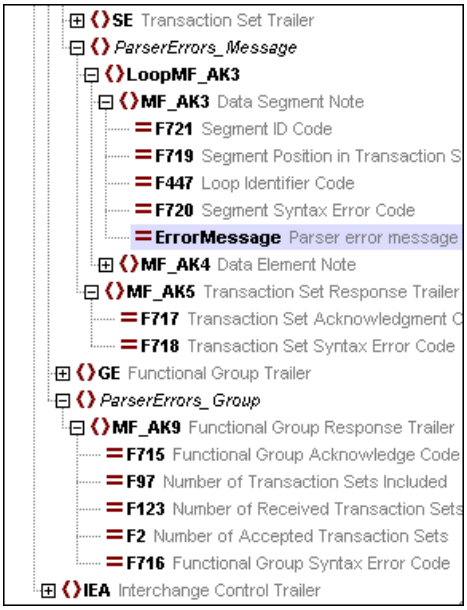
EDI 検証イベントのために構成することのできるカスタムアクション (**停止、無視** など) は、以下のとおりです

名前	説明	エラーの発生箇所
見つからないセグメント	必須セグメントが見つからないか、最低出現数未満の数です。	ソースコンポーネント
予期せぬセグメント	セグメントは定義されていますが、このメッセージには出現しません。	ソースコンポーネント

名前	説明	エラーの発生箇所
認識できないセグメントID	この仕様では定義されていないセグメントを発見しました。	ソースコンポーネント
見つからないグループ	必須グループが見つからないか、最低出現数未満の数です。	ソースコンポーネント
予期せぬファイルの終端	見つからないデータのため、インスタンスの解析を行うことができません。	ソースコンポーネント
見つからないフィールドまたはコンポジット	必須フィールドまたは複合要素が見つからないか、最低出現数未満の数です。	ソースまたはターゲットコンポーネント
セグメントまたはコンポジットの冗長なデータ	入力インスタンスに、構文では指定されていないデータが含まれています。	ソースコンポーネント
不要な繰り返し	セグメント/複合内に実際にあるフィールドの数が、指定された最大数を超えました。	ソースまたはターゲットコンポーネント
無効なフィールド値	数値フィールドに無効な文字が含まれています。	ソースコンポーネント
無効な日付	日付フィールドに無効な文字が含まれているか、月または日の値が無効です。	ソースコンポーネント
無効な時間	時間フィールドに無効な文字が含まれているか、時または分の値が無効です。	ソースコンポーネント
数値のオーバーフロー	指定されたドメインを数値が超えてしましました。このエラーは生成されたコードでのみサポートされます。	ソースコンポーネント
短すぎるデータ要素	データ要素の長さが指定された最低値を下回っています。	ソースまたはターゲットコンポーネント
長すぎるデータ要素	データ要素の長さが指定された最大値を超えてします。	ソースまたはターゲットコンポーネント
無効なコードリスト値	コードリストの値が指定された値のセットにマッチしません。	ソースまたはターゲットコンポーネント
論理エラー	論理エラーが発生しました。	ソースまたはターゲットコンポーネント
実装にて「使用されない」となっているデータ要素が存在	入力ファイル内に存在する要素はHIPAA 構成ファイル(`maxOccurs="0"`)では許可されていません。	ソースコンポーネント
入力ファイルが未解析	入力が完全に解析されませんでした。	ソースコンポーネント

検証結果とX12 Acknowledgment

ソースEDI コンポーネントに対して、検証結果(レポート)はEDI コンポーネントをベースとする「仮想」アイテムへ配置されます(ParserError_Message ならびにParserErrors_Group 以下)。これらのアイテムはX12 997 または 999 Acknowledgment にて使用することができます。インターチェンジの状態を報告するX12 Acknowledgment ファイルを作成する方法については次を参照してください:X12 997 Functional Acknowledgement の生成 とX12 999 Implementation Acknowledgment の生成)。



X12 またはHIPAA.X12 コンポーネントに関しては、EDI 検証設定 ダイアログボックス上で使用することのできる検証アクションがあり、(上記参照)以下を意味します :

停止 : レポートメッセージを生成すること無くマッピングの実行を停止します

報告して拒否ならびに**報告して受理** : EDI コンポーネントの Parser_Errors_Message ならびに Parser_Errors_Group アイテムに情報を与え、更に処理を継続することができます。

拒否するか、**受理**するかにより、Functional Group Acknowledge Code F715 ならびに Transaction Set Acknowledgment Code F717 に含まれる値が以下のように変化します :
"Rejected" (拒否) を表す 'R'
"受け入れられましたが、エラーが発生しました"を表す 'E' エラーはメッセージウィンドウにも表示されます。

無視 : 特定のエラーを無視します。Parser_Errors_Message ならびにParser_Errors_Group アイテムに情報が表示されることはありません。

7.5.5 ASC X12

ASC X12 はドキュメントのインターチェンジのための業界の標準です。MapForce は以下のバージョンをサポートします : 3040、3050、3060、3070、4010、4020、4030、4040、4041、4042、4050、4051、4052、4060、5010、5011、5012、5020、5030、5040、5050、6010、6020、6030、および 6040。

MapForce のデフォルトの ASC X12 バージョンは 6040 です。

ASC X12 コンポーネントには、EDI 検証 設定ダイアログボックス内で選択された設定によりEDI パーサーエラー情報やデータが書き込まれる「仮想」ノードがあります (EDI コンポーネントの検証を参照)。 X12 997 Functional Acknowledgement はすべてのX12 ドキュメントから生成することができます。

ASC X12 に関しての詳しい情報は、http://www.x12.org/ を参照してください。

7.5.5.1 ASC X12 検証ルール

ANSI X12 ドキュメントの検証時には以下の項目がチェックされます：

- ISA ならびにIEA セグメントが存在するか。
- ISA/I01 内に、法的承認情報修飾子が含まれるか。
- ISA/I03 内に、法的セキュリティー情報修飾子が含まれるか。
- 2 つのISA/I05 セグメントに法的インターチェンジID 修飾子が含まれるか。
- ISA/I08 内に整形式の日付値が含まれるか。
- ISA/I09 内に整形式の時刻値が含まれるか
- ISA/I13 内に法的 boolean 値が含まれるか。
- ISA/I14 内に法的インターチェンジ使用インジケーターが含まれるか。
- ISA/I12 とIEA/I12 内に含まれる値が同一のものか。
- IEA/I16 内に含まれる値が、インターチェンジ内に含まれる関数グループの正しい数か

各関数グループに対して、以下の項目がチェックされます：

- マッチするGS とGE のペアが存在するか。
- GS/373 内に整形式の日付値が含まれるか。
- GS/337 内に整形式の時刻値が含まれるか。
- GS/28 ならびにGE/28 内に含まれる値が同一のものか。
- GE/97 内に含まれる値が、関数グループ内に含まれるメッセージの正しい数か。

各メッセージに対して、以下の項目がチェックされます：

- マッチするST とSE のペアが存在するか。
- ST/143 内に正しいメッセージ識別子が含まれるか。
- ST/329 とSE/329 内に含まれる値が同一のものか。
- SE/96 内に含まれる値が、メッセージ内に含まれるセグメントの正しい数か。

7.5.5.2 ASC X12 自動データ補完ルール

ASC X12 ファイルを生成する際、MapForce は、("自動データ保管" または "自動補完" ルールとして説明されている)ルールに従い 自動的に特定のフィールドまたは値を生成する可能性があります。この振る舞いを無効化するには、EDI コンポーネント設定内のチェックボックスの **不足しているフィールドの自動補完** のチェックを解除してください。

EDI/X12 の自動データ補完により、以下の項目が保証されます：:

- インターチェンジレベルで、ISA とIEA のペアが存在する。
- GS またはGE の一方が存在する場合、もう一方のID も存在する。
- メッセージレベルにて、少なくとも 1 つのST/SE ペアが存在する。
- ISA/I01 ならびにISA/I03 が存在する。データが含まれていない場合、00 が挿入されます。
- ISA/I02 ならびにISA/I04 が存在する。データが含まれていない場合、10個の空白が挿入されます。
- ISA/I05 セグメントが存在する。データが含まれていない場合、"ZZ" が挿入されます。
- ISA/I08 が存在する。データが含まれていない場合、EDI フォーマット内にある現在日時が挿入されます。
- ISA/I09 が存在する。データが含まれていない場合、EDI フォーマット内にある現在時刻が挿入されます。
- ISA/I65 が存在する。データが含まれていない場合、繰り返しセパレーターが挿入されます。
- ISA/I11 が存在する。データが含まれていない場合、ユーザー定義設定から得られたインターチェンジ制御バージョン番号が挿入されます。EDI コンポーネント設定 の 拡張 | インターチェンジ制御バージョン番号」を参照してください。

- ISA/I12 の存在。
- ISA/I13 が存在する。データが含まれていない場合、確認を要求設定が使用されます。EDI コンポーネント設定の 拡張 | 確認を要求」を参照ください。
- That ISA/I14 の存在。データが含まれない場合、P が挿入されます。
- ISA/I15 が存在する。データが含まれていない場合、ユーザー定義設定から得られた複合セパレーターが挿入されます。EDI コンポーネント設定 の 設定 | 複合セパレーター」を参照ください。
- IEA/I16 が存在する。データが含まれていない場合、インターチェンジ内にある関数グループの数が計算され、挿入されます。
- IEA/I12 が存在する。データが含まれていない場合、ISA/I12 がコピーされます。

メモ： このセクションにて言及されなかったフィールドが（自動データ補完により）挿入または作成されることはありません。正しい値を自動的に確定することはできません。例えば、（EDI コンポーネント内に）ターゲット親要素のA と、それらの子アイテムx、y、z があり、そのうちy が必須アイテムとなっている場合、子要素の "y" がターゲットコンポーネントへマップされた時だけ、親要素のA が作成されます。

潜在的に存在する関数グループがチェックされます：

- GS/373 が存在する。データが含まれていない場合、EDI フォーマットにある現在の日付が挿入されます。
- GS/337 が存在する。データが含まれていない場合、EDI フォーマットにある現在の時刻が挿入されます。
- GE/97 が存在する。データが含まれていない場合、関数グループ内にあるメッセージの数が計算され、挿入されます。
- GE/28 が存在する。データが含まれていない場合、GS/28 から得られた値がコピーされます。

メッセージのチェックにより、以下の項目が保証されます：

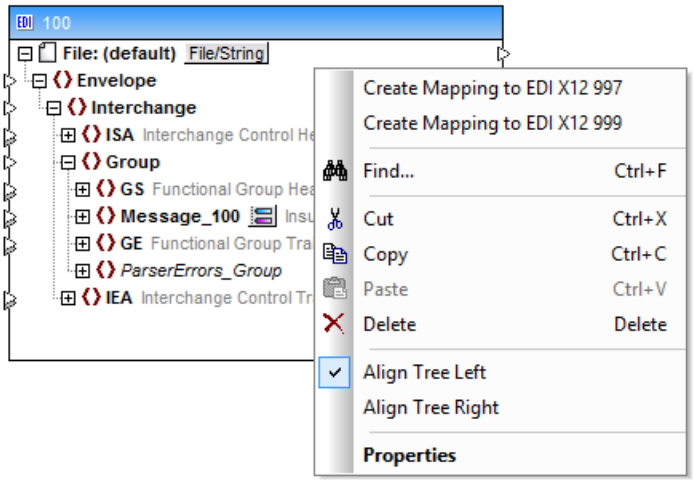
- ST/143 が存在する。データが含まれていない場合、メッセージの名前が挿入されます。
- SE/96 が存在する。データが含まれていない場合、メッセージ内にあるセグメントの数が計算され、挿入されます。
- ST/329 ならびにSE/329 が存在する。SE/329 内にデータが含まれていない場合、ST/329 から得られた値がコピーされます。

7.5.5.3 X12 997 Functional Acknowledgemen の生成

X12 997 (Functional Acknowledgement) は、EDI インターチェンジの状態を報告します。ドキュメントの処理中に発生するすべてのエラーは、この中に報告されます。MapForce は、自動的にメインのマッピングエリア内にX12 997 コンポーネントを生成し、自動的に必要なコネクタを作成します。

EDI 997 Functional Acknowledgement の生成：

1. **「挿入」**メニューから、**「EDI」**をクリックします。X12 コンポーネントをマッピングに追加します。インスタンスファイルとして選択するように問われると、X12 997 確認を作成するためのファイルを参照します。
2. ソースEDI コンポーネントを右クリックし、**「EDI X12 997 へマッピングを作成する」**を選択します（このコマンドは、X12 コンポーネントのためのみに有効化されています）。



これは、EDI 997 マッピングコンポーネントを作成し、X12 997 確認を生成するために必要なアイテムを自動的に接続します。を参照してください、**<Documents>\Altova\MapForce2017\MapForceExamples** フォルダー内の **X12_To_XML_Order.mfd** マッピングデザインファイルを参照してください。このマッピングから生成された確認ファイルを確認するには、EDI 997 コンポーネントの **「プレビュー」**（ ）ボタンをクリックし、そして **「出力」** タブをクリックします。

```
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
```

X12 997 Acknowledgement ファイルを保存する：

- **「出力」** タブをクリックして、そして **「生成された出力を保存」**（ ）ツールバーボタンをクリックします。マッピングからプログラムコードを生成する場合、コンポーネント設定 ダイアログボックスの "Output EDI File" フィールド内にファイル名を入力します（次を参照 :EDI コンポーネント設定）.

7.5.5.4 X12 999 Implementation Acknowledgment の生成

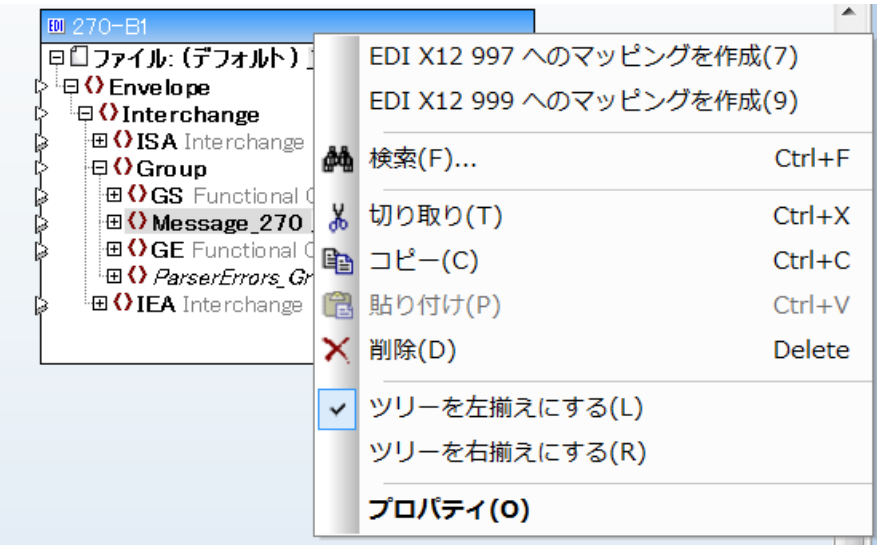
X12 999 (Implementation Acknowledgment) は、X12 非準拠、および、アプリケーションエラーを報告します。997 Functional Acknowledgement のスーパーセットであるため、X12 または HIPAA 実装ガイド構文の必要条件をベースにしたトランザクションを受理、または、拒否するために、999 を 997 の代わりに使用することができます。

ドキュメントの処理中に発生するエラーは、このファイルに報告されます。（例えば、"必要とされるセグメントが不足しています"、"必要とされるデータ要素 が不足しています"、"この実装内でコード値は使用されていません"、など）。MapForce は、自動的にマッピング上に X12 999 コンポーネントを生成、および、必要なマッピングの接続を自動的に作成することができます。

X12 999 Implementation Acknowledgment の生成：

1. **「挿入」** メニューから、**「EDI」** をクリックします。X12 または HIPAA コンポーネントをマッピングに追加します。インスタンスファイルとして選択するように問われると、X12 999 確認を作成するためのファイルを参照します。

2. ソースEDI コンポーネントを右クリックし、 **EDI X12 999 へマッピングを作成する**」を選択します (このコマンドは、X12 とHIPAA コンポーネントのためのみに有効化されています)。



これは、EDI 999 マッピングコンポーネントを作成し、X12 999 実装確認を生成するために生成するために必要なアイテムを自動的に接続します。生成された確認ファイルをプレビューするには **出力」**タブをクリックします。マッピングに複数のターゲットコンポーネントが存在する場合、最初に **プレビュー」**( ) ボタンをクリックし、そして タブをクリックします。

```
1   ISA+00+          +00+          +ZZ+ReceiverID    +ZZ+SenderID      +110502+1533+I+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
```

999 Acknowledgement ファイルを保存する:

- **出力」**タブをクリックし、そして **生成された出力を保存」**( ) ツールバーボタンをクリックします。マッピングからプログラムコードを生成する場合、コンポーネント設定 ダイアログボックスの 出力 EDI ファイル」フィールドをクリックします (次を参照 :EDI コンポーネント設定)。

7.5.6 HIPAA X12

HPAA X12 は、米国保険社会福祉省により制定された最新バージョンの電子ヘルスケアレコード標準で、EDI X12 バージョン 5010 に従ったかたちで、保険会社、医療機関、雇用主間における電子的な医療データのやり取りを行うために使用されます。

MapForce では最新のリリースである HPAA 実装仕様 (TR3) の A2 がサポートされています。それ以前のリリースは ZIP ファイルとして Altova Web サイトからダウンロードいただけます。

HIPAA コンポーネントは通常の ANSI X12 コンポーネントと類似しており、MapForce では以下の トランザクションがサポートされます :

X12 名	メッセージ名

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)"

MapForce における標準的な X12 メッセージ処理方法との違い：

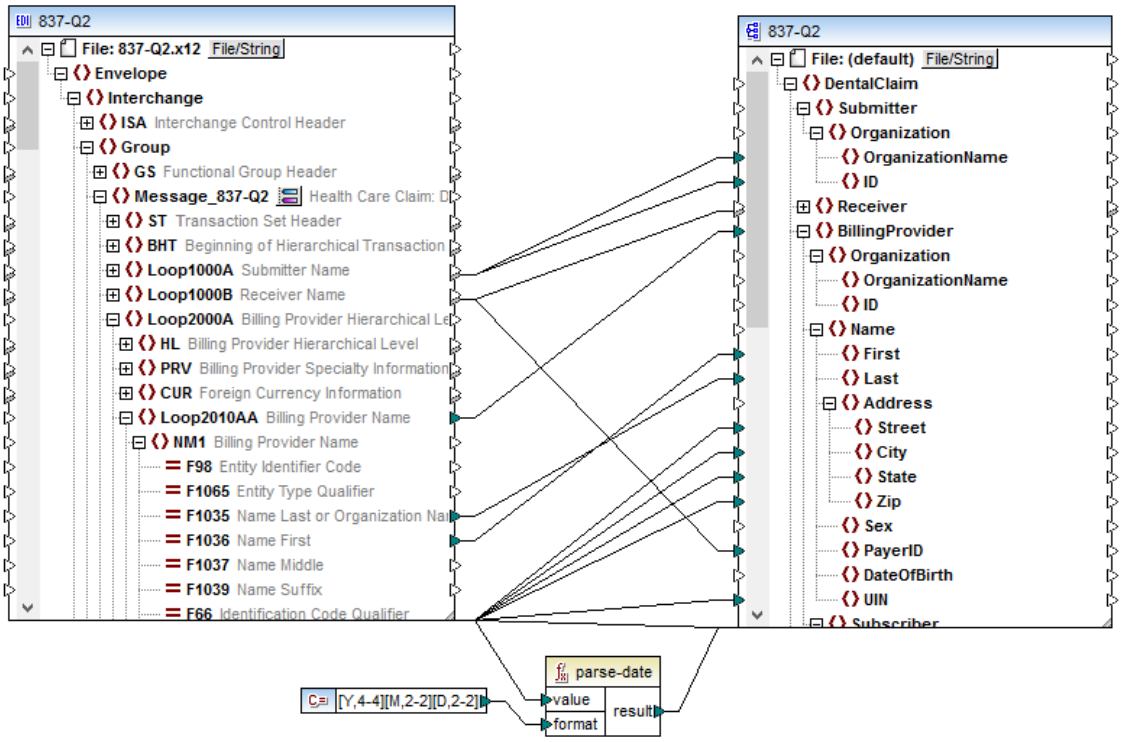
- HL セグメントの階層を自動的に管理
- (837 メッセージにおける) 浮動構造のサポート
- さうなるフィールドの自動補完と検証

MapForce では、標準的な X12 における既存の 999 や 997 Functional Acknowledgement と以た、HIPAA X12 999 Implementation Acknowledgment の自動生成をサポートしています (次を参照：X12 999 Implementation Acknowledgment の生成)。

HPAA コンポーネントでは、インターチェンジ (EDI コンポーネント) における複数のメッセージがサポートされています。

7.5.6.1 サンプル：HIPAA X12 から XML へのマッピング

このサンプルは、"837 Q2 Health Care Claim: Dental" トランザクションを含むファイルの XML ターゲットファイルへマップする方法について説明しています。このマッピングサンプルは以下のパスで見つけることができます：**<Documents>\Altova\MapForce2017\MapForceExamples\HIPAA_837D.mfd**。



HIPAA_837D.mfd

マッピングは下に示されているステップどおり作成することができます。

ステップ 1：ソース HIPAA X12 ファイルをマッピングに追加する

1. **挿入**」メニューから、「**EDI**」をクリックします。
2. "HIPAA.X12" コレクションを選択し、メッセージ型 "837Q2 Health Care Claim: Dental" の横のチェックボックスを選択します。
3. インスタンスファイルを提供するように問われると、次のパスから使用することのできるソースファイルを開きます：**<Documents>\Altova\MapForce2017\MapForceExamples\837-Q2.x12**.
4. プロンプトされると、必要に応じて、新規に追加されたHIPAA X12 コンポーネントの設定を調整します（詳細に関しては、次を参照してください：EDI コンポーネント設定）.

ステップ 2：マッピングにターゲットスキーマを追加する

1. **挿入**」メニューから、「**XML スキーマ/ファイル**」をクリックし、**<Documents>\ Altova \ MapForce2017\ MapForceExamples** ディレクトリから**837-Q2.xsd** ファイルを開きます。
2. サンプルXML ファイルを提供するように問われると、**「スキップ」**をクリックします。ターゲットドキュメントのルートとして**DentalClaim** を選択します。

ステップ 3：日付をフォーマットするために関数を追加し、マッピング接続を描く

ソースHIPAA ファイル内では、DMG セグメントの日付は、`19490501` のフォーマットで表示されています。このフォーマットをターゲットXML ファイル`1949-05-01` に適応するために変更するには、日付の変換関数を以下のように追加します：

1. **parse-date** 関数をライブラリウィンドウからマッピングにドラッグします。関数の詳細に関しては、次を参照してください: 関数と作業する
2. 「**挿入**」メニューから、「**関数**」をクリックします。定数として **[Y,4-4][M,2-2][D,2-2]** を入力します。
3. 以下の接続を描きます：

接続元	接続先	目的
関数	**parse-date** 関数の format 入力アイテム	日付が解析されるフォーマットに従い関数を提供します。
F1251	**parse-date** 関数の value 入力アイテム	フォーマットする実際の値を関数に提供します。
parse-date 関数の result 出力アイテム	XML ターゲットの DateOfBirth 入力アイテム	DateOfBirth XML ノードに対する関数に返された値を提供します。

4. サンプルのマッピングに表示されるように残りのマッピング接続を描きます（リストするには、接続の数が多すぎます。基本的には、各接続はHIPAA ソースファイルからターゲットXML ファイルに値をコピーします。マッピングの接続に関する基本の上は、次を参照してください: 接続と作業する）。

7.5.6.2 HIPAA トランザクションのマッピング時に考慮する点

HL セグメント階層構造

HL セグメントが自動的に生成され、ユーザーが手動でマッピングを行う必要はありません。必要な場合、値のマッピングを行うこともできます。

自動補完

HIPAA 変換に使用される構成ファイルには数多くのコードが含まれており、フィールドの自動補完に使用されます。自動補完は状況に応じて変化するもので、フィールドの位置により異なる値が含まれることもあります。コンポーネント設定ダイアログボックスにある「空のフィールドを自動補完」チェックボックスのチェックを外すことで、自動補完を無効にすることができます (次を参照: EDI コンポーネント設定).

検証

状況を意識するという意味で、検証と自動補完は共通しています。検証エラーは、X12 検証と同様に扱われます(次を参照: ASC X12 検証ルール)。

構成ファイルの編集

HIPAA 構成ファイルは、X12 構成ファイルよりも新しいスキーマバージョンを使用します（次を参照：EDI 構成スキーマ)。このため、HIPAA 構成ファイルは、X12 構成ファイルに類似しています：

- フィールドレベルのコード値
- 条件
- 補完フラグ
- 「使用されない」データ要素

コード値

フィールドインスタンスには、検証や自動補完に使用される内蔵のコードリストを含めることができます。値は手動で追加、削除することができます。(次を参照: データ要素のコードの値の変更の可能性).

```
<Data ref="F365" info="Communication Number Qualifier" nodeName="F365_1">
<Values>
        <Value Code="EM" />
        <Value Code="FX" />
        <Value Code="TE" />
</Values>
</Data>
```

条件

条件は値とともに使用されます。繰り返されている複数のセグメントやグループ (ループ) が存在しうるため、特定の条件に従ってそれらを識別する必要があります。条件には値を含める必要があり、値が含まれない場合、特定のグループを識別することは不可能になります。

例えば、Loop1000 が複数回繰り返されており、各インスタンスがセマンティック上で異なる意味を持つような状況が考えられます。

どれが目的の要素なのか識別するため、以下のコードにより NM1 セグメントのフィールド F98 にある "41" という値コードを持った Loop100A が MapForce で識別されます。このような条件に合致する要素が存在しない場合、Loop1000A が発見されることはありません。

```
<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>
(… )
```

条件コードに値が含まれる場合、条件コードがターゲットコンポーネントにて自動生成されます。

メソッドにより生成された構成ファイルにより、繰り替えされたセグメント/ループのシーケンスにて条件の値が一意であることが保証されます。

条件やその中にある値を編集する場合、値によりターゲットが一意に定まることを考える必要があります。

補完プラグ

ファイルレベルとともに、自動補完の調整を行うことができます。新たに用意された "Completion" 要素に記述される 3つの属性により、以下にあるような定義が行われます：

- singleConditions ？ターゲットコンポーネント内にある全てのフィールドに対して、単一の条件を自動補完します。
- singleValues ？ターゲットコンポーネント内にある全てのフィールドに対して、単一の値を自動補完します。
- HL ？ターゲットコンポーネント内にある HL セグメントにて、適切なフィールドを生成します (全 HL セグメントが対象)。

  (Where "1" means true)

```
<Message>
  <MessageType>837-Q2</MessageType>
  <Completion singleConditions="1" singleValues="1" HL="1" />
  <Description>Health Care Claim: Dental</Description>
(… )
```

「使用されない」データ要素

標準的な X12 トランザクションでは表示されるオプション要素のいくつかは、HIPAA にて省略されます。これらのオプション要素は MapForce コンポーネントにて隠されます。

検証エンジンのチェックにより、これら使用されないフィールドがソースファイルに存在しないことが保証されます。これらのフィールドは隠されており、ターゲットコンポーネントにマッピングすることができないため、ターゲット出力ファイルに表示されることもありません。

構成ファイルではこれらのフィールドに対して値が 0となっている maxOccurs 属性が定義されます。必要な場合は、maxOccurs 属性の値を手動で変更することにより、特定のフィールドを隠す または表示する) ことができます。

```
<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" />
```

検証のスコープ
MapForce ではセマンティックな検証はサポートされていません。状況に応じて意味が異なるフィールドはオプションとして扱われます。

7.5.7 HL7 Version 2

HL7 は医療アプリケーションのデータ交換のための業界の標準で、 "Health Level Seven" の省略形です。MapForce はバージョン 2.2 から 2.6 をサポートします。

MapForce 内のデフォルトの HL7 バージョンは 2.6 です。

追加 HL7 V2.2 - V2.5.1 XML スキーマと構成ファイルのための個別のインストーラーが Altova Web サイト MapForce コンポーネントのページ (http://www.altova.com/ja/components_mapforce.html) で使用することができます。

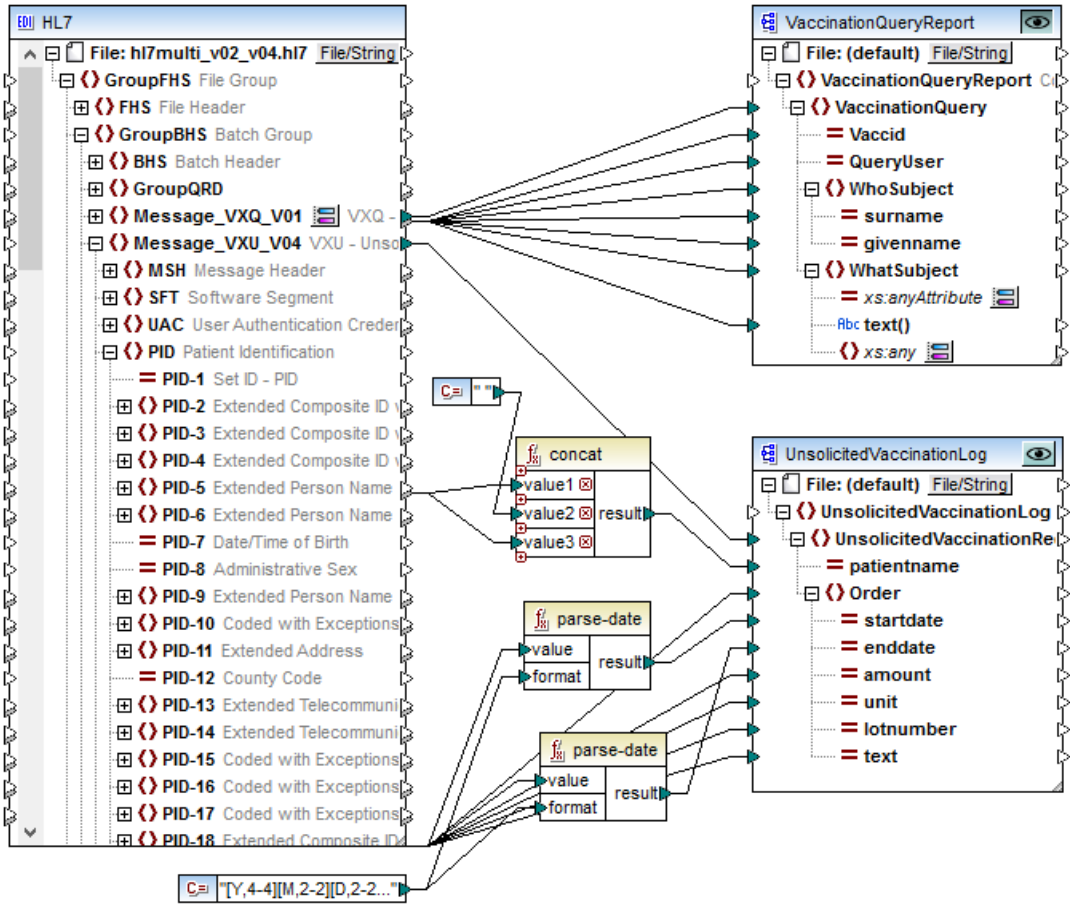
XML-ベースの HL7 バージョン 3.x は、XML スキーマ コンポーネントを使用して、MapForce 2017 内でサポートされています。

MapForceExamples プロジェクトは、HL7 V2.6 から HL7 V3 XML ファイルへマップするサンプル (HL7V260_to_HL7V3.mfd) が含まれています。

HL7 に関しての詳しい情報は、http://www.hl7.org/ を参照してください。

7.5.7.1 サンプル HL7 から XML へのマッピング

このサンプルは、2つの HL7 メッセージ ("Query for vaccination record" と "Unsolicited vaccination record update") を含むファイルの2つの個別の XML ターゲットファイルへマップする方法について説明しています。このマッピングサンプルは以下のパスで見つけることができます :**<Documents>\Altova\MapForce2017\MapForceExamples\HL7_MultiMessageTypes.mfd**。



HL7_MultiMessageTypes.mfd

マッピングは下に示されているステップどおり作成することができます。

ステップ 1：ソース HL7 ファイルをマッピングに追加する

1. **挿入**」メニューから、「**EDI**」をクリックします。
2. "HL7" コレクションを選択し、メッセージ型 "VXQ - Query for vaccination record" と "VXU - Unsolicited vaccination record update" の横のチェックボックスを選択します。
3. インスタンスファイルを提供するように問われると、次のパスから使用することのできるソースファイルを開きます：**<Documents>\Altova\MapForce2017\MapForceExamples\hl7multi_v02_v04.hl7**.
4. プロンプトされると、必要に応じて、新規に追加されたHL7 コンポーネントの設定を調整します（詳細に関しては、次を参照してください：EDI コンポーネント設定）。

ステップ 2：マッピングにターゲットスキーマを追加する

1. **挿入**」メニューから、「**XML スキーマ/ファイル**」をクリックし、**<Documents>\ Altova \ MapForce2017\ MapForceExamples** ディレクトリから**VaccinationQueryReport.xsd** ファイルを開きます。
2. サンプルXML ファイルを提供するように問われると、「**スキップ**」をクリックします。

マッピングコンポーネントとして**UnsolicitedVaccinationLog.xsd** を追加するには、同じ手順を踏みます。

ステップ 3：関数を追加し、マッピング接続を描く

ソースHL7 ファイル内では、日付は、`YYYYMMDD` のフォーマットで表示されています。このフォーマットをターゲットXML ファイル`YYYY-MM-DD` に適応するために変更するには、日付の変換関数を以下のように追加します：

1. **`parse-date`** 関数をライブラリウィンドウからマッピングにドラッグします。関数の詳細に関しては、次を参照してください[関数と作業する]
2. **「挿入」**メニューから、**「定数」**をクリックします。定数として`[Y,4-4][M,2-2][D,2-2]` を入力します。
3. 以下の接続を描きます：

接続元	接続先	目的
定数	**`parse-date`** 関数の`format` 入力アイテム	日付が解析されるフォーマットに従い関数を提供します。
`RXA-3`	**`parse-date`** 関数の`value` 入力アイテム	フォーマットする実際の値を関数に提供します。
`parse-date` 関数の`result` 出力アイテム	XML ターゲットの`startdate` 入力アイテム	`startdate` XML ノードに対する関数に返された値を提供します。

新規 **`parse-date`** 関数と同じステップを踏み、`RXA-4` の値を変換するために、ターゲットXMLの`enddate` アイテムにコピーします。

2番目のターゲットXML 内の`patientname` アイテムは、ソースHL7 ファイルのの2つのフィールドを結合することにより作成されます：`FN-1` (Surname) and `XPN-2` (Given Name). これを達成するには、ライブラリウィンドウから、**`concat`** 関数をマッピングにドラッグします。また、2つの名前を分けるスペース文字を提供する定数を追加します。

サンプルのマッピングに表示されるように残りのマッピング接続を描きます（リストするには、接続の数が多すぎます。基本的には、各接続はソースHL7 ファイルからターゲットXML ファイルに値をコピーします。マッピングの接続に関する基本の上は、次を参照してください[接続と作業する]）。

ターゲットXML コンポーネントにより生成された出力をプレビューするには、**「プレビュー」**（ ）ボタンをクリックし、**「出力」**タブをクリックします。

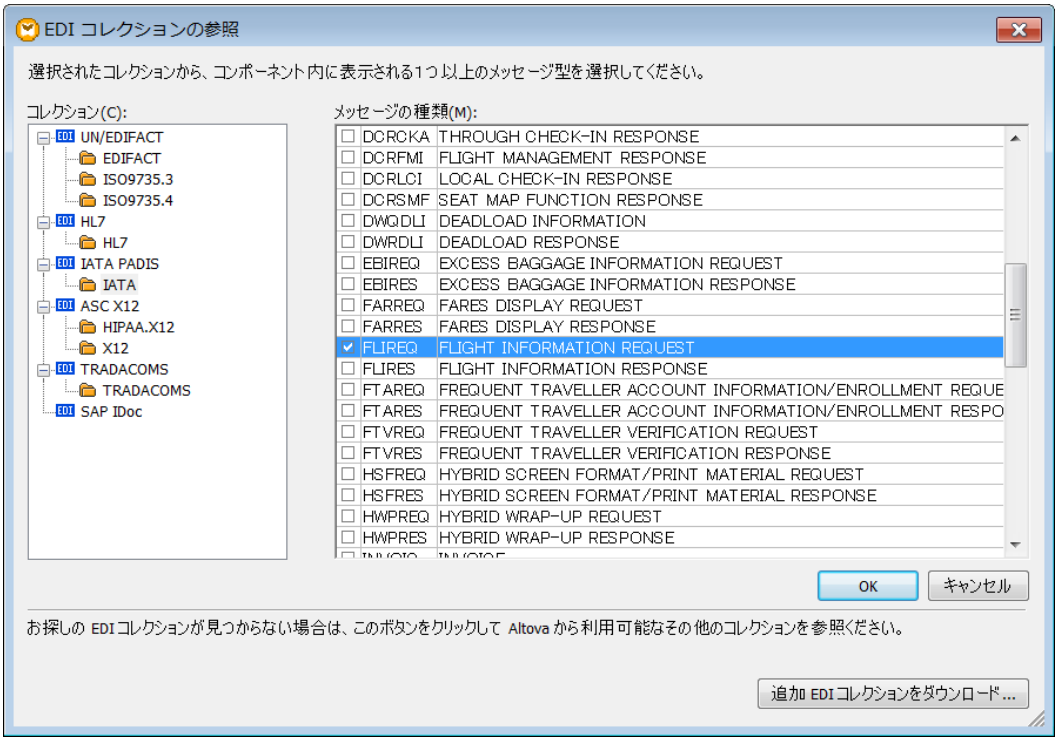
7.5.8 IATA PADIS

PADIS (Passenger and Airport Data Interchange Standards) は、EDIFACT (ISO 9735) 構文を使用したメッセージのセットです。

- MapForce は、現在メッセージヘッダーとトレーラーセグメントを使用したメッセージのみをサポートしています。
- MapForce は、IATA.00.1 からIATA.08.1. までのコレクションをサポートしています。

IATA PADIS ファイルをマッピングコンポーネントとして追加する：

1. **「挿入」**メニューから、**「EDI」**をクリックします。
2. "IATA PADIS" から "IATA" コレクションをクリックし、必要なメッセージ型の横のチェックボックスを選択します）。

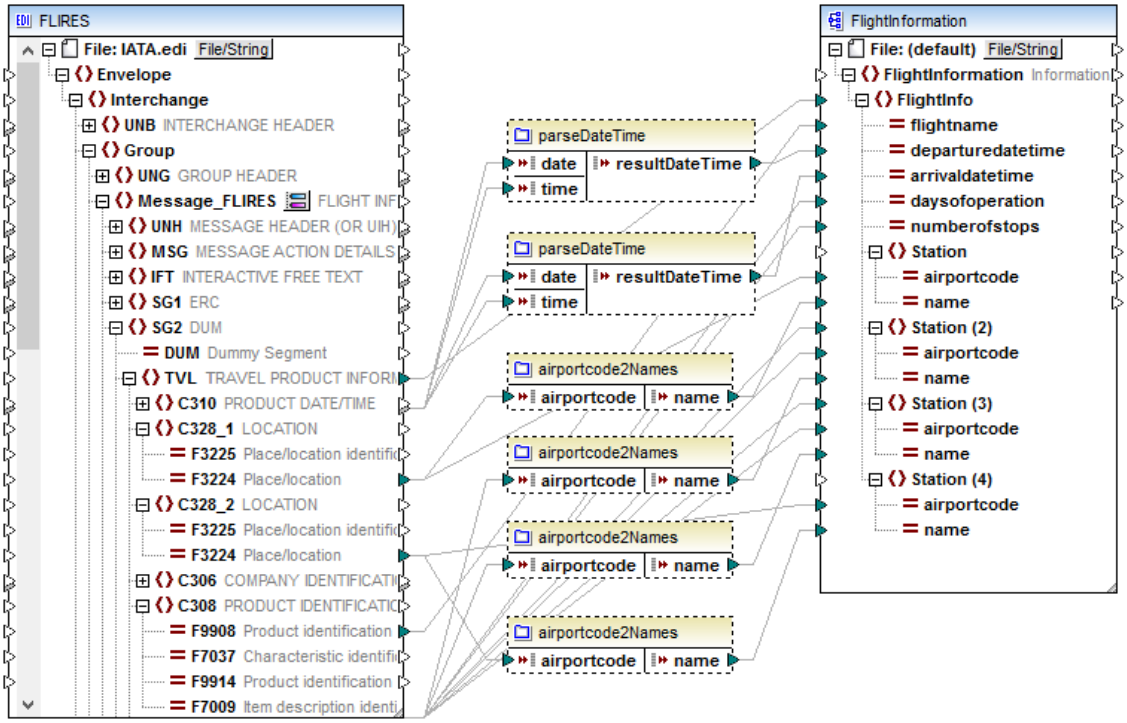


3. インスタンスファイルを提供するようにプロンプトされると、**「参照」**をクリックして、ソースIATA ファイルをマッピングソースとして使用するために開きます。IATA PADIS コンポーネントにマッピングする場合、**「スキップ」**をクリックします。click.

IATA コンポーネントがマッピングに追加され、マッピングの接続を描くことができます。 IATA コンポーネントの設定を変更するには、右クリックして、コンテキストメニューから **「プロパティ」** を選択します (次も参照 : EDI コンポーネント設定)。

7.5.8.1 サンプル IATA を XML にマッピングする

<Documents>\Altova\MapForce2017\MapForceExamples フォルダー内の **IATA_FlightInformationReport.mfd** ファイルは、 XML ターゲットファイルにIATA PADIS ファイルをマッピングする例を示しています。



IATA_FlightInformationReport.mfd

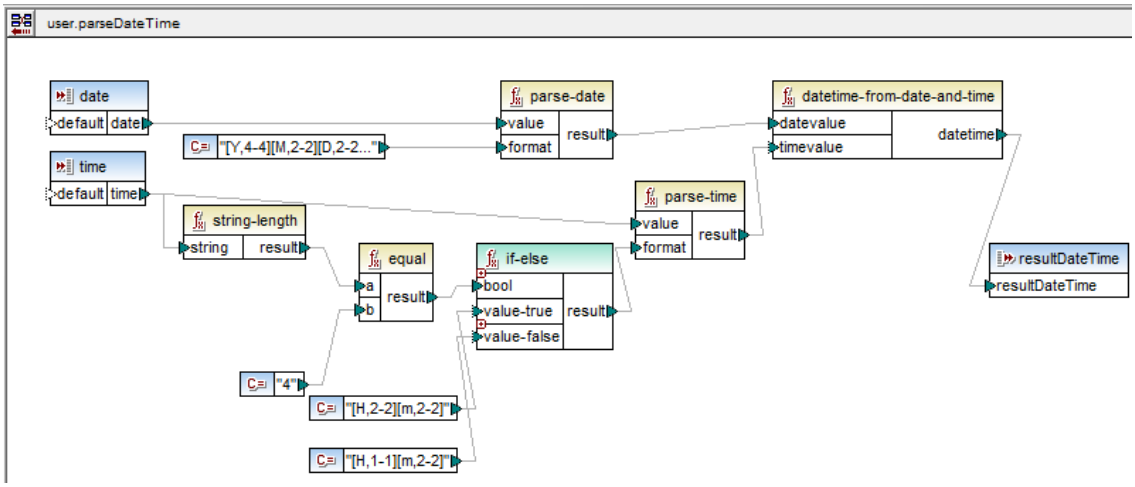
このマッピングは、FLIRES メッセージ、ターゲット XML コンポーネント、および 2つのユーザー定義関数 **parseDateTime** と (複数回使用されている) **airportcode2Names** を含むソース IATA コンポーネントにより構成されています。

ソース、または、ターゲットコンポーネント、および他の設定からインスタンスファイルへの読み込み、および書き込みを確認するには、コンポーネントを右クリックして、**「プロパティ」を**選択してください (次も参照 : EDI コンポーネント設定)。

各 ユーザー定義関数を確認するには、ダブルクリックします。メインのマッピングに戻るには、マッピングエリアの左上にある **「メインのマッピングに戻る」** ( ) ボタンをクリックしてください。詳細に関しては、次を参照してください : ユーザー定義関数。

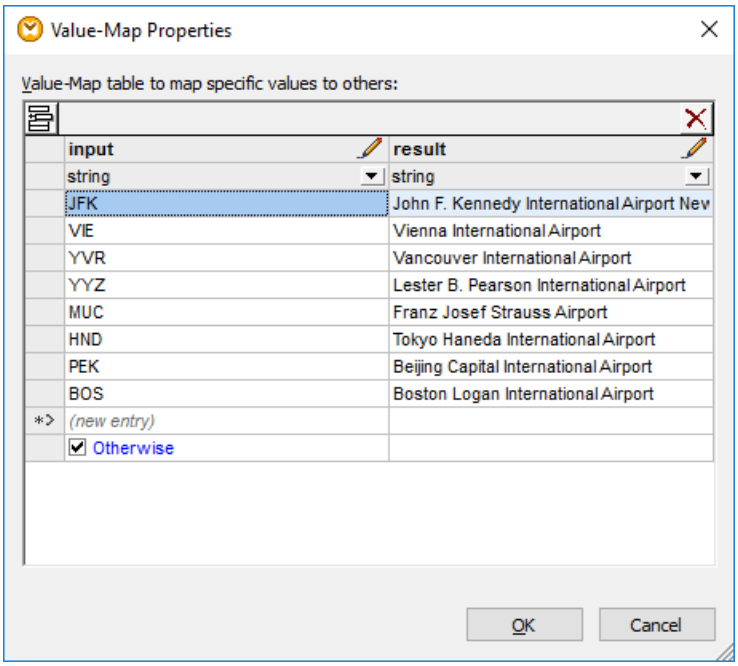
このマッピング内では、ユーザー定義関数には以下の目的があります :

1) **parseDateTime** 関数は、ターゲット XML ファイル内で必要とされる `xs:dateTime` 値を処理するために、引数として、IATA 日付と時刻を日付および時刻要素から受け取ります。このゴールを達成するために、MapForce コアライブラリ内の多種の関数を活用することができます。



parseDateTime 関数の構造

2) **airportcode2Names** 関数は、F3223 または F3224 フィールド値の引数として受け取られ、空港の名前を返します。これを達成するには、関数には、以下のように空港名に各空港のコードをマップする Value-Map コンポーネントが含まれている必要があります：



Value-Map コンポーネントの構成

Value-Map コンポーネントに関する詳細は、次を参照してください Value-Map を使用する。

ターゲット側では、XML コンポーネントは、"ノードの複製" 呼ばれる技術を使用しています。具体的には、ノード `Station` がマッピングコンポーネント上で複数回複製されることを意味します。これにより、同じターゲットアイテムに複数のソースアイテムからマップ値をマップすることができます（次を参照：入力の複製）。

マッピングにより生成された出力をプレビューするには、**「出力」** タブをクリックします。

7.5.9 SAP IDoc

SAP IDocs (中間伝票) ドキュメントは、SAP と非 SAP アプリケーション間におけるデータのやり取りに使用されます。ドキュメントは、異なるシステム間における中間データとして機能します

IDoc は以下のような構造を持ちます：

- **コントロールレコード:** 送信者や受信者、メッセージや IDoc の種類といった IDoc に関する制御情報が含まれます。制御レコードは、あらゆる種類の IDoc にて同じような書式になります。
- **データセグメント:** セグメントに含まれる実際のデータや、その他メタデータのヘッダー、セグメントの番号や種類、データが含まれるフィールドといったデータが含まれます。
- **ステータスレコード:** ドキュメントの現在ステータス (現在処理されているステージ) と、これから処理を行うステージに関する情報が含まれます。ステータスの書式は、全種類の IDoc にて共通しています。

Port 定義にある Version 番号にて、通信を行なっているシステムが定義されます。バージョン間における主な違いは、各要素にて使用される名前の長さと、拡張の使用に関するものです。SAP R3 バージョン 4.X では、これまでのバージョンには無かった長い名前 (そして拡張) がサポートされます。

ポート バージョン 1: リリース 2.1 ならびに 2.2.
ポート バージョン リリース 3.0、3.1、R/2 システム
ポート バージョン 3: リリース 4.x (デフォルト値)

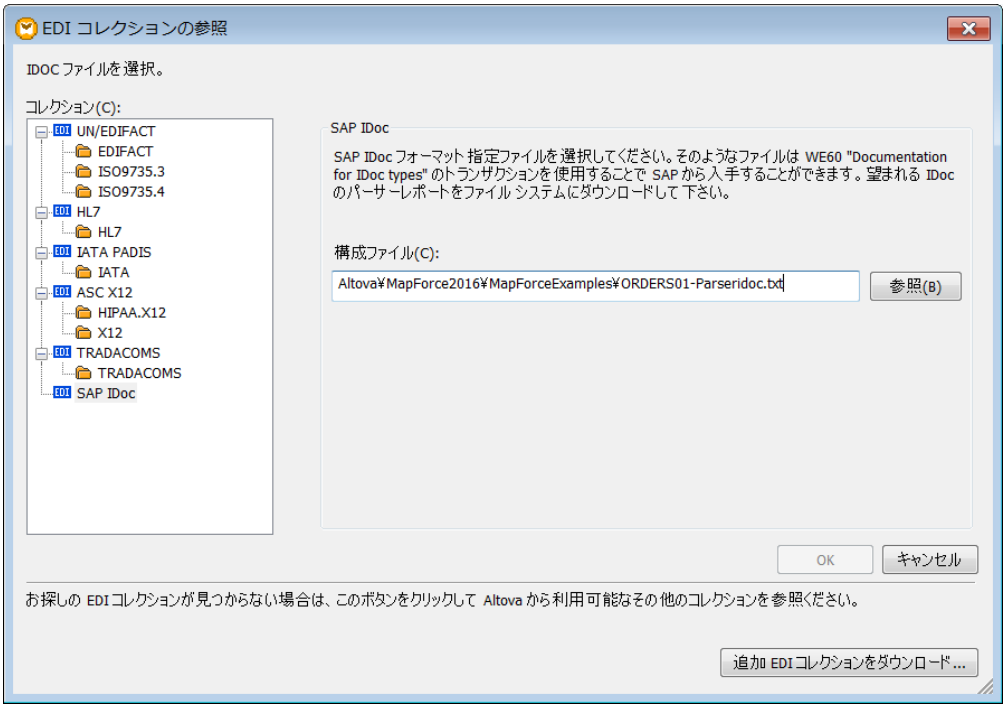
MapForce において、IDoc コンポーネントはメッセージの種類に対して30文字、IDoc の種類に対して30文字、セグメントフィールドに対して27文字が割り当てられた固定長のファイルとして扱われます。

..\MapForceExamples フォルダー内にある **<Documents>\Altova\MapForce2017\MapForceExamples\IDoc_Order.mfd** ファイルでは、SAP IDoc から XML スキーマターゲットファイルへのマッピングサンプルを確認することができます。

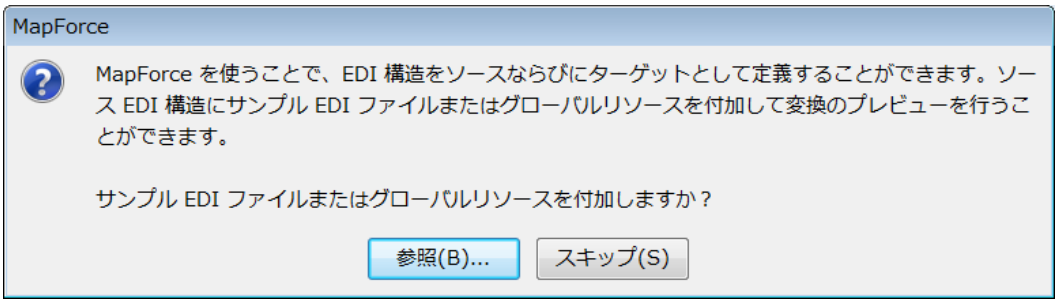
SAP IDoc ドキュメントをマッピングコンポーネントとして挿入する:

1. メニューオプションから **「入力 | EDI」** を選択します。EDI コレクションの参照 ダイアログボックスが表示されます。
2. リストボックスから **SAP IDoc** エントリーをクリックします。
3. **「参照」** ボタンをクリックして、("IDoc 定義ファイル" としても知られる) IDoc パーサーレポートを選択します .

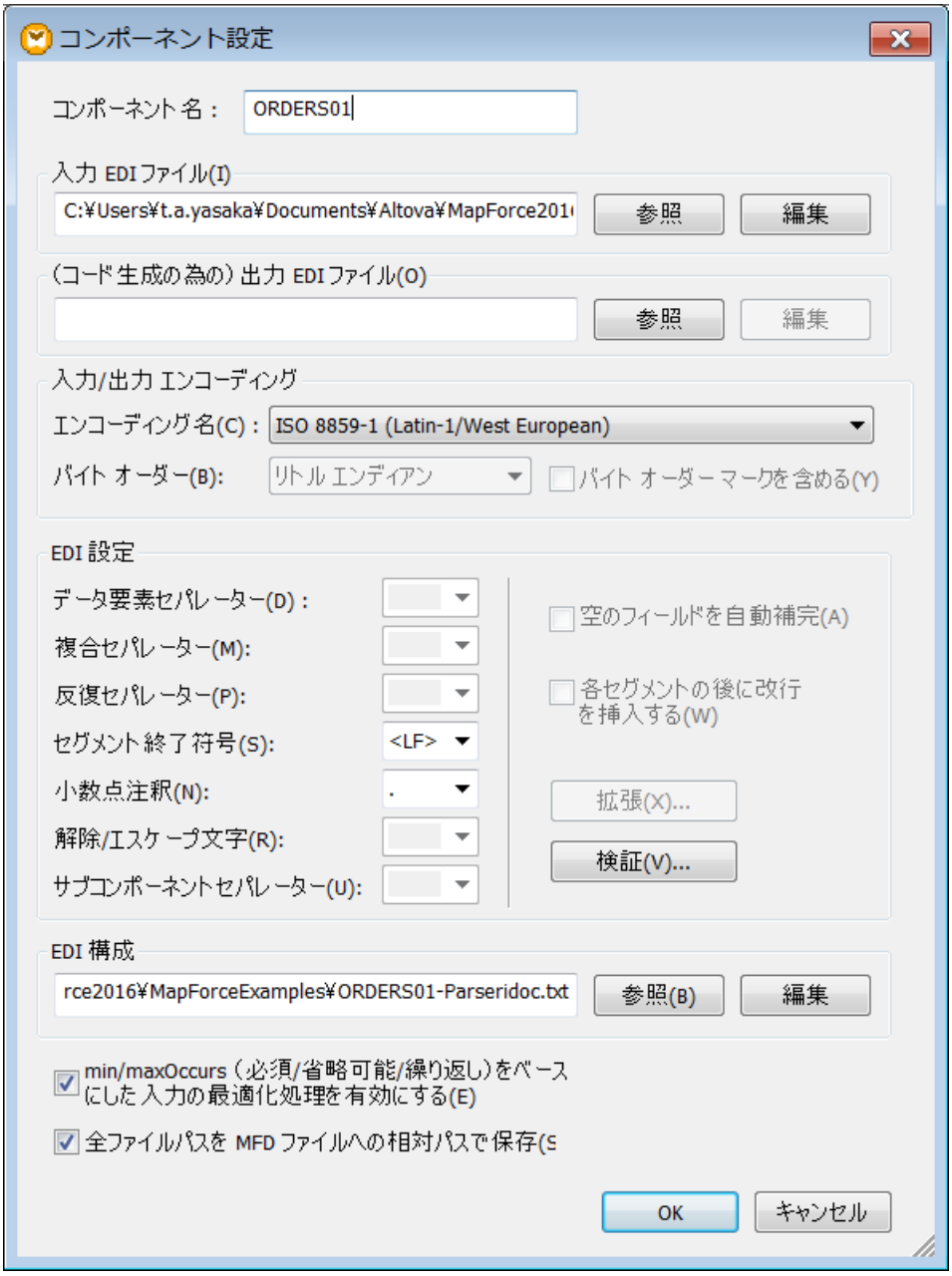
> IDoc パーサーレポートは、使用中の SAP システムから WE60 "ドキュメント for IDoc types" を使用することで得ることができます。IDoc_Order.mfd サンプルに対して構成ファイルの **ORDERS01-Parseridoc.txt <Documents>\Altova\MapForce2017\MapForceExamples** フォルダーに収められています。



4. **「OK」**をクリックして、EDI ファイルを選択します



5. IDoc ファイルからデータをマップする場合、 **「参照」**をクリックして、データを提供するIDoc (*.idoc) ファイルを選択します。サンプルIDoc ファイルは、MapForce Examples フォルダーにあります (**<Documents>\Altova\MapForce2017\MapForceExamples\ORDERS.idoc** を参照する)。その他の場合は、 **「スキップ」**をクリックします。
6. コンポーネント設定ダイアログボックスが表示されます。これにより、マッピングに追加する前にIDoc コンポーネントの設定をレビューすることができます。必要な場合この設定を後に変更することができます。 (コンポーネント設定を変更する を参照する)。



7. 「OK」をクリックして、コンポーネント設定ダイアログボックスを閉じます。IDoc コンポーネントが挿入されます。

7.5.10 TRADACOMS

TRADACOMS (Trading Data Communications) は英国内で使用される小売業の電子データ交換の標準です。1982 に最初のEDI 標準として英国の経済と産業のために導入されました。 UN/EDIFACT 標準と多くの類似点を有することから、TRADACOMS は、EDIFACT の先駆の1つでありますが、メッセージ内で異なる構造を使用します。

MapForce は、ベースTRADACOMS 仕様を 1993年 1月に、現在ではGS1 UK (https://www.gs1uk.org) として知られる、英国 Article Numbering Association (ANA) により出版された "TRADACOMS Manual of Standards for Electronic Data Interchange" 内で説明されているように実装し

ます。その他のTRADACOMS バージョンのために、MapForce を、新規メッセージの型、データ要素、およびコード値を構成ファイルを使用して処理するようにカスタム化することができます。

MapForce 内で、TRADACOMS フォーマットと他のサポートされるEDI フォーマットと同様に以下のように作業できます：

- TRADACOMS ファイルをMapForce によりサポートされる他の全てのデータフォーマットで (データソース または データターゲットとして) 両方の方向にマップすることが出来ます。 .
- (マッピングのプレビューに使用される、またはMapForce Server とFlowForce Server 実行に使用される) BUILT-IN 言語とコード生成 言語 (C++、C#、Java) 内でTRADACOMS フォーマットにデータをマッピング、またはTRADACOMS フォーマットからデータをマッピングすることができます。異なるマシン上で動作する FlowForce Server にマッピングをデプロイする場合、デプロイされたパッケージは、自動的に選択された TRADACOMS メッセージ型の構成とデータ要素により使用されたすべてのコードリストを含みます。同様に、生成されたコード内では、MapForce は、構成グループ セグメント、データ要素のためにクラスを生成します。
- 拒否される (または受け入れられる) または無視される レコードデータの変換を停止する検証イベントを柔軟に定義することができます。
- データの自動補完を有効化または無効化することができます。"自動的なデータの保管 " または "自動補完 ") が有効化されると、MapForce は、TRADACOMS ファイルの生成の際に、値を自動的にフィルします。
- メッセージの型、データ要素、およびコードリストを構成ファイルを使用し、グローバルまたはローカルでをカスタム化することができます (構成ファイルを参照)。
- 解析されたまたは生成されたファイルのエンコードを設定することができます。

以下のTRADACOMS ファイル型とメッセージの型がデフォルトでMapForce 内で使用することができます：

ファイル型	メッセージ型	バージョン
THE PRODUCT INFORMATION FILE	PROHDR (Product File Header)	8
	PROINF (Product Details)	8
	PROTLR (Product File Trailer)	8
THE PRICE INFORMATION FILE	PRIHDR (Price File Header)	8
	PRIINF (Price Details)	8
	PRITLR (Price File Trailer)	8
CUSTOMER INFORMATION FILE	CUSHDR (Customer Information Header)	8
	CUSINF (Customer Information Details)	8
	CUSTLR (Customer Information Trailer)	8
THE ORDER FILE	ORDHDR (Order File Header)	9
	ORDERS (Order Details)	9
	ORDTLR (Order File Trailer)	9
THE PICKING INSTRUCTIONS FILE	PICHDR (Picking Instructions File Header)	4
	PICKER (Picking Instructions File Details)	4
	PICTLR (Picking Instructions File Trailer)	4
THE DELIVERY NOTIFICATION FILE	DELHDR (Delivery File Header)	9
	DELIVR (Delivery Details)	9

ファイル型	メッセージ型	バージョン
	DELTLR (Delivery File Trailer)	9
DELIVERY CONFIRMATION FILE	DLCHDR (Delivery Confirmation Header)	5
	DLCDET (Delivery Confirmation Details)	5
	DLCTLR (Delivery Confirmation Trailer)	5
THE INVOICE FILE	INVFIL (Invoice File Header)	9
	INVOIC (Invoice Details)	9
	VATTLR (VAT Trailer)	9
	INVTLR (Invoice File Trailer)	9
THE CREDIT NOTE FILE	CREHDR (Credit Note File Header)	9
	CREDIT (Credit Note Details)	9
	VATTLR (File VAT Trailer)	9
	CRETLR (Credit Note File Trailer)	9
STATEMENT/REMITTANCE FILE	SRMHDR (Statement/Remittance Details File Header)	9
	SRMINF (Statement/Remittance Line Details)	9
	SRMTLR (Statement/Remittance Details File Trailer)	9
UPLIFT INSTRUCTION FILE	UPLHDR (Uplift File Header)	4
	UPLIFT (Uplift File Details)	4
	UPLTLR (Uplift File Trailer)	4
UPLIFT CONFIRMATION FILE	UCNHDR (Uplift Confirmation Header)	3
	UCNDET (Uplift Confirmation Details)	3
	UCNTLR (Uplift Confirmation Trailer)	3
THE STOCK SNAPSHOT FILE	SNPHDR (Stock Snapshot Header)	3
	SNPSTS (Stock Snapshot Details)	3
	SNPTLR (Stock Snapshot Trailer)	3
THE STOCK ADJUSTMENT FILE	SADHDR (Stock Adjustment Header)	3
	SADDET (Stock Adjustment Details)	3
	SADTLR (Stock Adjustment Trailer)	3
AVAILABILITY REPORT FILE	AVLHDR (Availability File Header)	4
	AVLDET (Availability Report Details)	4
	AVLTLR (Availability File Trailer)	4
GENERAL COMMUNICATIONS FILE	GENHDR (General Communications File	3

ファイル型	メッセージ型	バージョン
	Header)	
	GENRAL (General Communications Text)	3
	GENTLR (General Communications Trailer)	3
COMPLEX ORDER FILE	CORHDR (Complex Order File Header)	6
	CORDER (Complex Order Details)	6
	CORTLR (Complex Order Trailer)	6
THE ACKNOWLEDGEMENT OF ORDER FILE	ACKHDR (Acknowledgement File Header)	4
	ACKMNT (Acknowledgement Details)	4
	ACKTLR (Acknowledgement Trailer)	4
PRODUCT PLANNING REPORT FILE	PPRHDR (Product Planning Report Header)	2
	PPRDET (Product Planning Report Details)	2
	PPRTLR (Product Planning Report Trailer)	2
THE PAYMENT ORDER FILE	PAYHDR (Payment Order File Header)	3
	PAYINF (Payment Order Line Details)	3
	PAYTLR (Payment Order File Trailer)	3
THE DEBIT ADVICE FILE	DRAHDR (Debit Advice File Header)	3
	DRAINF (Debit Advice Line Details)	3
	DRATLR (Debit Advice File Trailer)	3
THE CREDIT ADVICE FILE	CRAHDR (Credit Advice File Header)	3
	CRAINF (Credit Advice Line Details)	3
	CRATLR (Credit Advice File Trailer)	3
THE EXCEPTION CONDITION FILE	EXCHDR (Exception Condition File Header)	3
	EXCINF (Exception Condition Line Details)	3
	EXCTLR (Exception Condition File Trailer)	3
LOCATION PLANNING REPORT FILE	LPRHDR (Location Planning Report Header)	2
	LPRDET (Location Planning Report Details)	2
	LPRTLR (Location Planning Report Trailer)	2

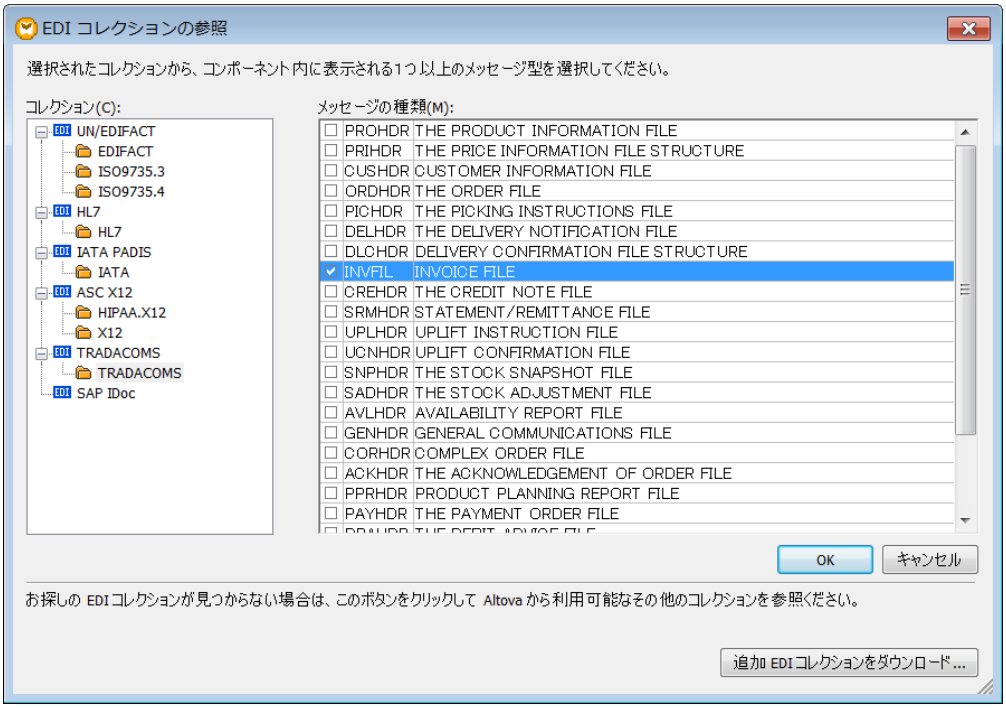
ファイル型	メッセージ型	バージョン
THE UTILITY BILL FILE	UTLHDR (Utility Bill File Header)	3
	UTLBIL (Utility Bill File Details)	3
	UVATLR (Utility Bill VAT Trailer)	3
	UTLTLR (Utility Bill File Trailer)	3

メモ: RSGRSG (調整メッセージ) バージョン 2 もサポートされています。このメッセージの型は全てのファイルの型で発生する可能性があります。

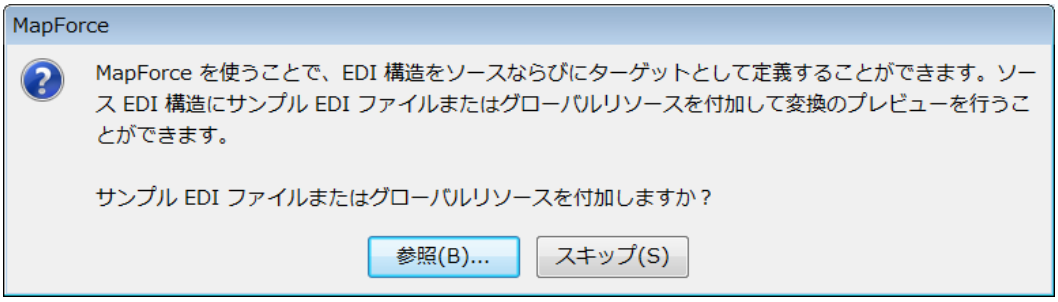
7.5.10.1 TRADACOMS ファイルをマッピングコンポーネントとして追加する

データマッピングソースとして、TRADACOMS ファイルを追加する:

1. 以下を行います:
   - o **挿入」**メニューから**EDI** をクリックします。
   - o **「EDI ファイルの挿入」**( ) ツールバーボタンをクリックします。
2. 「EDI コレクションの参照」ダイアログボックスから、**「TRADACOMS」**をクリックして、メッセージに含まれるメッセージの型を選択します。



3. **「OK」**をクリックします。インスタンスファイルを提供するようにプロンプトされます。



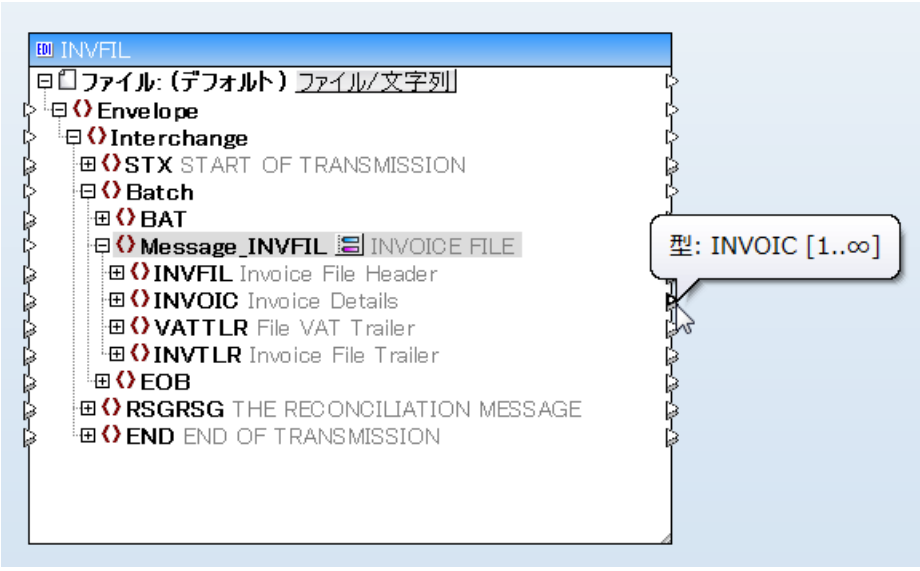
4. ソースTRADACOMS ファイルを後で指定する場合、**「スキップ」**をクリックします。それ以外の場合、**「参照」**をクリックしてTRADACOMS ファイルの開き方を選択します:
   - ローカルドライブまたはネットワークからファイルを開く場合、ファイルを参照して、**「開く」**をクリックします。
   - URL からファイルを開く場合、**「URL に切り替える」**をクリックします（詳細に関しては、URL からコンポーネントを追加するを参照してください）。
   - ファイルを以前グローバルリソースとして定義した場合、グローバルリソースとしてファイル開きます**「グローバルリソースに切り替える」**をクリックします。（Altova グローバルリソースを参照）。
5. コンポーネント設定ダイアログボックスを設定または必要な場合設定を変更します（EDI コンポーネント設定 を参照）。

データマッピングターゲットとしてTRADACOMS ファイルを追加する：

1. 上記のステップ1、2、を繰り返します。
2. インスタンスファイルを提供するようにプロンプトされると、**「スキップ」**をクリックします。

7.5.10.2 MapForce 内の TRADACOMS コンポーネント

MapForce では、MapForce EDI-に関連したコンポーネントに共通する全ての機能を保持しつつ、RADACOMS コンポーネントが、（任意のバッチ機能を使用して）TRADACOMS 伝送の構造をレプリケートします。



MapForce 内のサンプルTRADACOMS コンポーネント

コンポーネントの上のアイテム **「ファイル」** は、処理されている、または、コンポーネントにより生成された、TRADACOMS インターチェンジ ファイル名を表示します。 "デフォルト" は、まだファイルが割り当てられていないことを意味します。 **「ファイル / 文字列」** (File/String) ボタンは、すべての ファイルベースのMapForce コンポーネントに共通する高度な機能を、複数のファイルを動的に読み込み、処理するなどの他のオプションと共に表示しています (詳細に関しては、複数の入力または出力ファイルを動的に処理を参照してください)。

"エンベロープ" と "インターチェンジ" 構造 はすべてのMapForce EDI-に関連した コンポーネントに対してジェネリックです。適応できる場合、 MapForce コンポーネント内で複数のインターチェンジを処理できる機能を与えます。

STX とEND 構造は、TRADACOMS フォーマットに対して特有であり、"伝送の開始 " および "伝送の完了 " セグメントを個別に表します。

BAT とEOB 構造は、TRADACOMS フォーマットに特有です。これらは "バッチの開始 " および "バッチの完了 " セグメントをそれぞれ表します。このようなセグメントは、TRADACOMS インターチェンジ ファイルからデータをバッチで読み込むの際に、またはファイルをバッチで生成する際に適用することができます。同じ伝送内に複数のバッチが存在する可能性があり、 **「バッチ」** シーケンスは、BAT シーケンスの親として使用することができます。

Message_*Code* 構造 (上のサンプル内の **Message_INVFIL**) は、TRADACOMS " ファイル " に対応します。 **EDI メッセージの型の選択** ボタン ( ) は、ファイルの型を変更することのできるダイアログボックスを開きます (例 : "請求書ファイル"、 "注文ファイル" 等)。 既存のMapForce EDI 機能により、同じコンポーネント内に複数のファイル型を含むことができます。 しかしながら、 TRADACOMS 仕様は、各インターチェンジ伝送内で1つのファイルと1つのファイルの型を使用することを奨励しています。

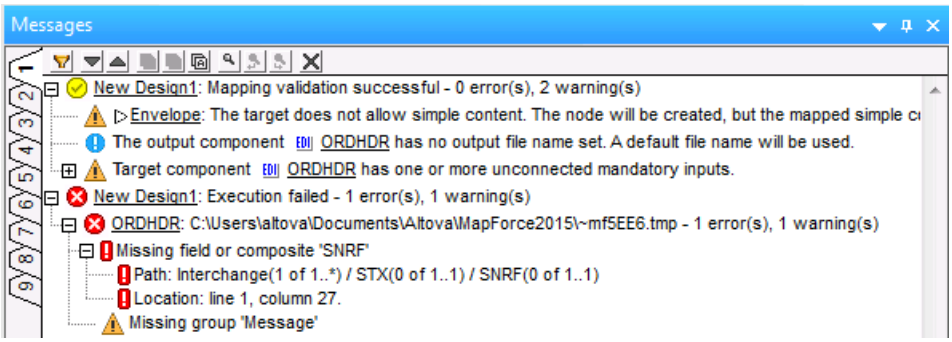
TRADACOMS ファイル (この例では、"請求書ファイル") には、ヘッダーメッセージが含まれます。1つまたは複数の詳細メッセージ、トレーラーメッセージが含まれています。上のコンポーネントでは INVFIL (請求書ファイルヘッダー)、INVTLR (請求書ファイルトレーラー)、VATTLR (ファイルVAT トレーラー) および、複数のINVOIC (請求書の詳細) メッセージが表示されています。

MapForce コンポーネントのそれぞれの両脇に、入力と出力コネクタ (小さな三角形) が表示されており、それぞれのデータ要素、または、セグメントをMapForce によりサポートされる他のデータ型、または、フォーマットに接続することができます。 **「表示 | ヒントの表示」** メニューでヒントの表示が有効化されていると想定して、コンポーネント上のアイテムにマウスをポイントすると、(発生の最高数および最小数など) それぞれに関する追加情報を確認することができます。

最後に、他の設定をコンポーネントヘッドをダブルクリック または コンポーネント設定ダイアログボックス を開くことにより変更することができます (EDI コンポーネント設定 を参照)。このダイアログボックスは、すべてのEDI-に関連した コンポーネント型のためのジェネリックなダイアログボックスであり、このため、オプションの一部はTRADACOMS に適用できない可能性があります。

7.5.10.3 検証と自動データ補完

TRADACOMS 構造からデータを読み込む、またはデータを書き込むマッピングを実行すると、MapForce は、構造データの検証を行い、TRADACOMS 仕様に従いチェックし、メッセージウィンドウ内に全ての検証エラーを表示します。

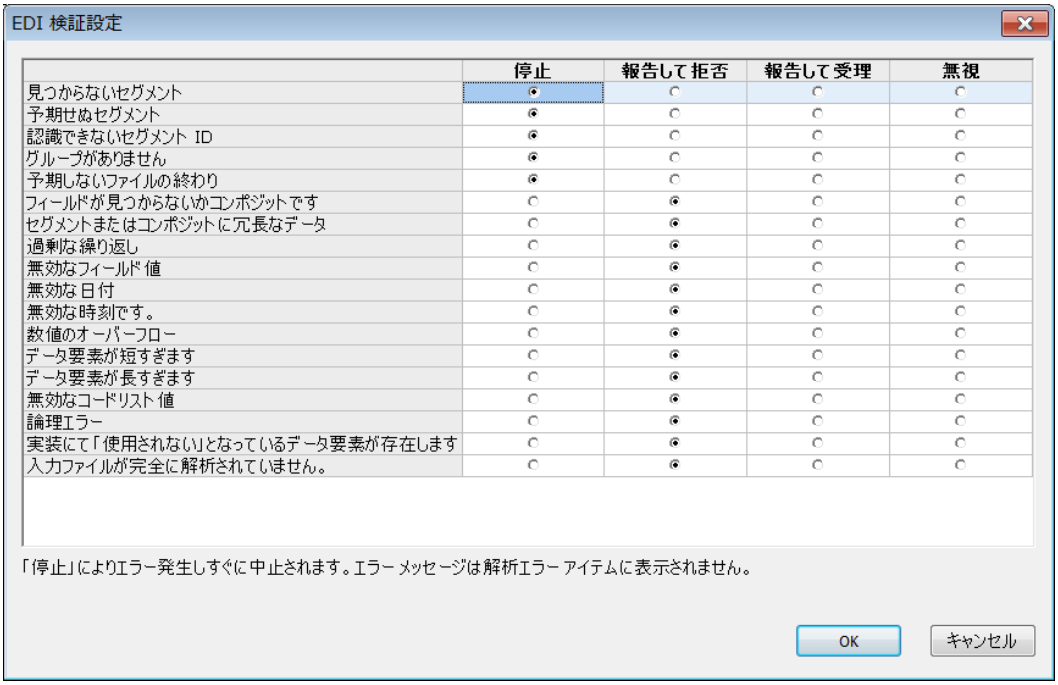


メッセージウィンドウwith 検証エラー

上のサンプルで示されるとおり、TRADACOMS に特定の検証メッセージ(情報、警告、とエラー)が、MapForce マッピングに共通するジェネリックな検証メッセージに加え表示されます。MapForce 検証全般に関しての詳しい情報は、マッピングの検証を参照してください。

以下の要素が、MapForce が解析された、または、生成されたTRADACOMS ファイルをどのように検証するかを決定します:

- 構成ファイル内出定義された検証 制約は、MapForce インストールフォルダー内で使用することができます (サブフォルダー**MapForceEDI\TRADACOMS**)。これらの構成ファイルは、一方、TRADACOMS 仕様のデフォルトの検証ルールを与えます。その一方で、カスタム化された必要性にTRADACOMS フォーマットを適用する手段を与えます。特に、構成ファイル内で定義されているデータ要素、セグメント、または コード値を変更することは可能ですが、検証の結果とマッピングの実行に影響を与えます。構成ファイルに関しての詳しい情報は、構成ファイルを参照してください。
- MapForce に内蔵の検証ロジックは、構成ファイルを使用せずに実行されるMapForce 内部データの整合性のチェックを含まれます。
- MapForce グラフィカルユーザーインターフェイスから定義したカスタム検証の設定 (EDI コンポーネントの検証 を参照)。TRADACOMS を含むEDI コンポーネントの現在のEDI 検証設定を確認するには、コンポーネント設定ダイアログボックスのコンポーネントのヘッダーをダブルクリックし、 **検証**」をクリックします。



TRADACOMS 構造に書き込む際、MapForce は、値が計算することのできる、または、事前に定義することのできる、データ要素のコンテンツを自動的にフィルします。この機能は"自動的な補完" (または "自動補完") と呼ばれます。この振る舞いを無効化するには、コンポーネント設定ダイアログボックスから、**不足しているフィールドを自動補完する**チェックボックスのチェックを解除します (EDI コンポーネント設定 を参照)。

以下のTRADACOMS 検証ルールに従い、MapForce は、(ファイル解析 または生成中に) 検証エラーを表示、または、(ファイル生成中に) 不足しているフィールドの自動補完を行います：

1. セグメントSTX (伝送の開始) およびEND (伝送の完了) は、存在しなくてはなりません。
2. STDS-1 が値 ANAA' を持つ場合、調整メッセージ (RSGRSG) が、伝送の終わり (END) の前に存在しなくてはなりません。それ以外の場合、調整メッセージ (RSGRSG) は、存在してはなりません。
3. STDS-1 が値 ANAA' を持つ場合：
   a. 調整メッセージ内のRSGA の値は、STX segment 内のSNRF の値と同じでなくてはなりません。
   b. 調整メッセージ内のRSGB の値は、STX segment 内のUNTO-1 の値と同じでなくてはなりません。
4. TRDT-1 は、日付 (YYMMDD) を含む必要があり、TRDT-2 は、伝送 (現在の日時) の時刻 (HHMMSS) を含む必要があります。
5. バッチヘッダー (BAT) が存在する場合、バッチトレーラー (EOB) も存在する必要があり、NOLI (バッチごとのメッセージ数) データ要素内でバッチ内のメッセージの数を確認することができます。
6. MSRF (メッセージ参照) データ要素 メッセージ内でヘッダー (MHD) は、1から開始される伝送内のメッセージの連番を含む必要があります。
7. NOSG (メッセージ内のセグメントの数) データ要素 メッセージ内でTrailer (MTR) は、MHD とMTR を含むセグメントの数を含む必要があります。
8. 存在する場合、調整メッセージ (RSGRSG) は、メッセージヘッダー とメッセージ トレーラーを含まない1つのセグメント (RSG) により構成されている必要があります。
9. END セグメント内のNMST (伝送内のメッセージの数) データ要素は、インターチェンジ (MHD セグメントの数) 内のメッセージの数を含む必要があります。
10. 全般的に、TRADACOMS 構造を読み取る際にMapForce は、インターチェンジ環境が型 "コンピューターからコンピューター" (または、in TRADACOMS 用語では、"インテリジェントなターミナルからインテリジェントなターミナル") であることを期待します。ですから、`MHD = 12 + ORDHDR :3` などのセグメントは余分な行頭と末尾のスペースが含まれるため、検証エラーがトリガーされます。
11. 文字列データは大文字である必要があります。TRADACOMS 出力を生成する際、MapForce は、文字列

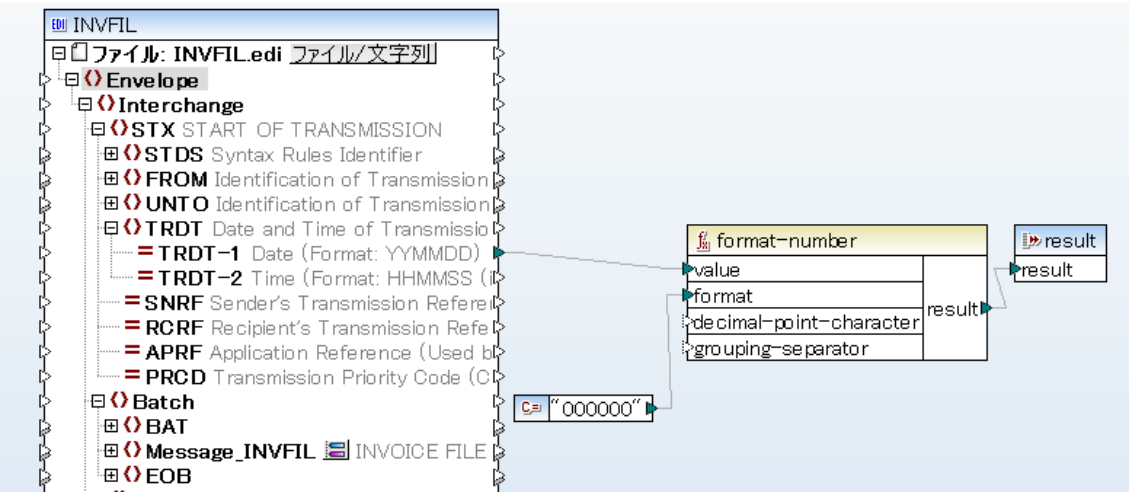
データを大文字に変換します。

更に、前述のように、構成ファイルで定義された検証ルールは、TRADACOMS の解析と生成に影響を与えます。

7.5.10.4 変換中に先頭の 0 (ゼロ)を保持する

TRADACOMS 仕様では、TRDT-1 とTRDT-2 (伝送の日時)データ要素のデータ型は、"小数"として定義されています。これは、デフォルトでは、TRDT-1 またはTRDT-2 フィールドからデータをマップすると、これらのフィールドの先頭のゼロが変換中トリムされ、期待されない結果を生成するということを意味します。これは、"小数"と定義されており、値を文字列として扱うように定義されている他のアイテムでも同様の結果が出る可能性があります。

core | format-number MapForce 関数を適用することにより、MapForce に数値フィールドを文字列として扱うように命令することができます(故に、先頭のゼロを保持することが可能になります)。下のサンプルでは、ソースファイル内のTRDT-1 アイテムの値は、"020312" です。(小数点の変換により)このアイテムの通常の出力は、期待されない結果である "20312"です。ですから、先頭のゼロを保持するために、**format-number** 関数がマッピングに追加されます。(関数との作業方法に関する詳しい情報は、関数と作業を参照してください)。



format-number 関数を使用して先頭の 0を保持する

上に示されるとおり、関数には次の2 つの入力引数があります:

1. フォーマットされる値。(この場合、 "020312")。
2. フォーマットマスク "000000"。

関数の出力をテストするには、この例は単純型出力コンポーネントを使用します(単純型出力を参照)。 **出力**」ボタンをクリックすると、マッピングの出力は期待される(関数の結果と同じ) "020312" となります。

7.5.10.5 構成ファイル

デフォルトでは、MapForce 構成ファイルと基となるMapForce 処理ロジックがTRADACOMS 標準 (931)のベースバージョンを反映します。ですから、このTRADACOMS バージョンを使用する場合、TRADACOMS 構成ファイルのカスタマイズは必要ありません。しかしながら、既存の構成への変更をチェックすることは可能です。標準TRADACOMS 構成ファイルに加えられた変更は、MapForce がTRADACOMS 構造を解析する、または、読み込む振る舞いに、直ちに影響が出ます。

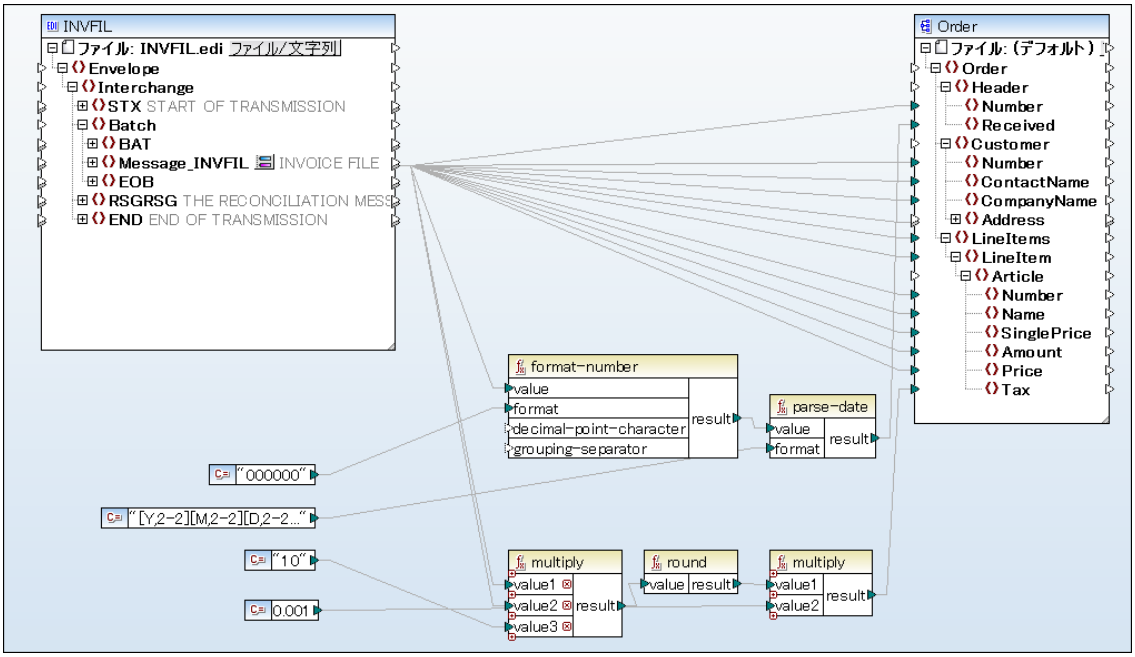
TRADACOMS に適用することのできるMapForce 構成ファイルは、以下のディレクトリに保管されます **:C:**

\Program Files\Altova\MapForce2017\MapForceEDI\TRADACOMS。TRADACOMS 構成ファイルに変更を加える場合、TRADACOMS ディレクトリの兄弟ディレクトリにコピーを作成します。例えば、 "TRADACOMS1" ディレクトリのパーミッションを "読み取り専用 "にせず、カスタムの必要条件に応じてファイルを編集します。すべての構成ファイルは、XML 構文に準じますので、XML エディターを使用してファイルを変更することが奨励されます。

ファイル	目的
EDI.Collection	**EDI.Collection** ファイルは、現在のTRADACOMS 構成ディレクトリ内のすべてのメッセージのリストを含みます (例 :**C:\Program Files\Altova \MapForce2017\MapForceEDI\TRADACOMS**)。 コレクションファイルにリストされるメッセージは、マッピングに TRADACOMS コンポーネントを (**入力 | EDI**」を使用して追加すると表示される "EDI コレクションの参照 " ダイアログボックス 内で選択のために使用できます。 ですから、このダイアログボックス内でメッセージの型を抑制するには、コメントアウトを使用するか、EDI.Collection ファイルから対応する XML 要素を削除してください。 例 :以下のリンクのコメントアウトは LPRHDR メッセージ型 の使用を不可能にします : `<!--<Message Type="LPRHDR" File="LPRHDR.Config" Description="LOCATION PLANNING REPORT FILE"/>-->`
Envelope.Config	MapForce 内でTRADACOMS コンポーネントの構造を定義します。(すなわち、要素が**Envelope** シーケンスにより囲まれています)。このファイルは、MapForce により内部的に処理されたスキーマに対して検証され、編集されてはなりません。
Tradacoms.Codelist	このファイルは TRADACOMS コードと含まれている可能性のある値を定義します。入力 /出力 TRADACOMS ファイルの検証にも使用されます。必要な場合、このファイルにカスタム コードリストを追加することができます。
Tradacoms.Segment	このファイルは、TRADACOMS ファイルのセグメント、複合、および、フィールド名を定義し、TRADACOMS ファイルを解析する際に使用されます。このファイルに加えられる変更は、グローバルにカスタマイズすることができ、全てのセグメントとメッセージに適用できます。
[Message].Config	単一のTRADACOMS メッセージの構造を定義します。このファイルへの変更はローカル (インライン) カスタマイズとして扱われます。

7.5.10.6 サンプル TRADACOMS 請求書をXML に変換する

このサンプルは、TRADACOMS 請求書ファイルのデータをXML フォーマットに変換するマッピングデザインの作成方法について説明します。 このサンプルで作成されたマッピングデザインファイルは、以下のパスで参照することができます : **<Documents>\Altova\MapForce2017\MapForceExamples\TRADACOMS_Invoice.mfd** 。



TRADACOMS_Invoice.mfd

マッピングを以下を目標とします：

1. ソースTRADACOMS ファイル (**NVFIL.edi**) を既存のスキーマ(**Order.xsd**) に対して有効なXML ファイルに変換します。MapForce Examples フォルダー (**<Documents>\Altova\MapForce2017 \MapForceExamples**) 内にソースTRADACOMS ファイルとスキーマファイル (**Order.xsd**) があります。
2. TRADACOMS ファイル内の日付には、`decimal` フォーマットがあり、ターゲットXML ファイル内の `xs:dateTime` フォーマットに変換される必要があります。
3. 注文のオーダー内の各ラインアイテムに対して、税金は、3桁の小数点を持つ小数の値まで計算される必要があります。このサンプルのスコープのために、税金は以下の式に従い計算されると想定します：

```
 Tax = round(LEXC * VATP * 10)
* 0.001
```

`round` は最も近い整数で値を四捨五入します。

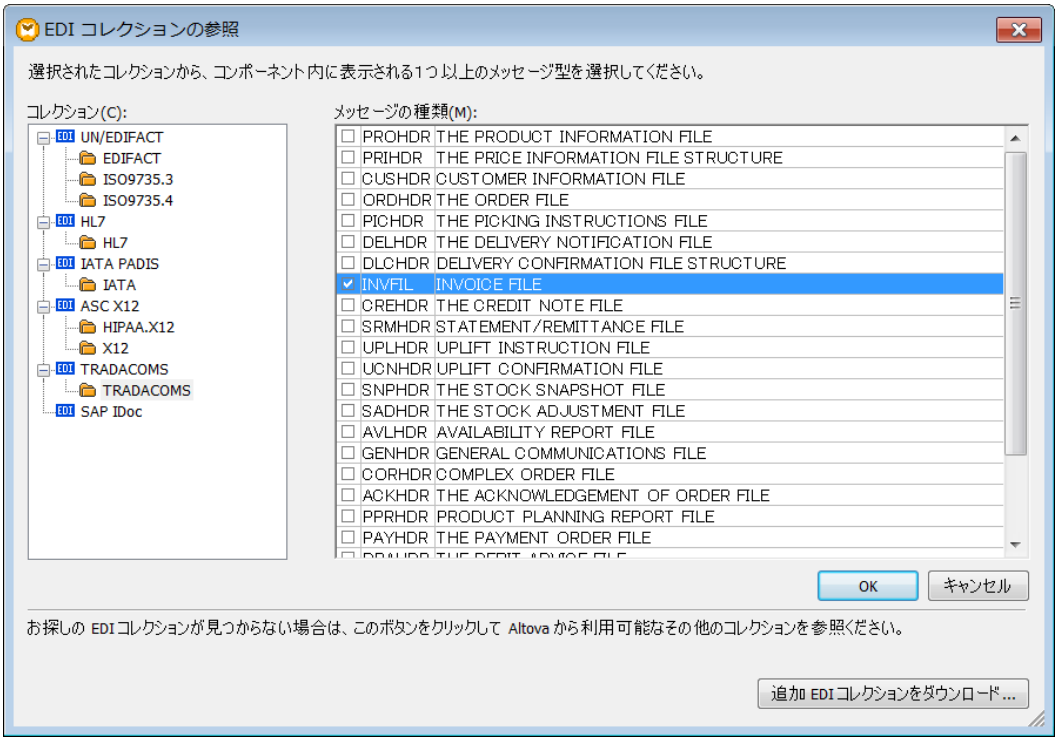
この目的を達成するには、以下の手順を踏みます：

1. ソースとターゲットコンポーネントをマッピングエリアに追加し、その間にマッピング接続を描きます。
2. 日付関数を使用して、日付を必要とされるフォーマットに変換します。
3. 数学関数を使用して、消費税を計算します。

ステップ 1：マッピングエリアにコンポーネントを追加する

ソースTRADACOMS コンポーネントを追加します：

1. **「挿入」** メニューから **「EDI」** をクリックします。
2. TRADACOMS コレクションフォルダーをクリックして、メッセージの型の下から、**INVOICE FILE** を選択します。



3. サンプルファイルを提供するようにプロンプトされると、MapForce Examples フォルダー (**<Documents>\Altova\MapForce2017\MapForceExamples**) 内の**INVFIL.edi** を参照します。
4. コンポーネント設定を入力するようにプロンプトされると、デフォルトの値をそのままにし、**「OK」**をクリックします。

ターゲット XML コンポーネントを追加する:

1. **「挿入」**メニューから**「XML スキーマ/ファイル」**をクリックします。
2. サンプルファイルを提供するようにプロンプトされると、**「スキップ」**をクリックします。(これはターゲットコンポーネントであるため、ファイルが生成されるため、既存のファイルを参照する必要はありません。)
3. ルート要素を選択するようにプロンプトされると、"Order" を選択します。

このサンプルでは、ソースがソース TRADACOMS ファイルからターゲット XML ファイルに以下の通りマップされます:

ソース (TRADACOMS)	ターゲット (XML)
INVN	/Order/Header/Number
IVDT*	/Order/Header/Received
CIDN-2	/Order/Customer/Number
CADD-1	/Order/Customer/ContactName
CNAM	/Order/Customer/CompanyName
CADD-2	/Order/Customer/Address/Street
CADD-3	/Order/Customer/Address/City
CADD-5	/Order/Customer/Address/ZIP
CADD-4	/Order/Customer/Address/State

ソース (TRADACOMS)	ターゲット (XML)
Group_IRF**	/Order/LineItems
Group_ILD**	/Order/LineItems/LineItem
SPRO-1	/Order/LineItems/LineItem/Article/Number
TDES-1	/Order/LineItems/LineItem/Article/Name
AUCT-1	/Order/LineItems/LineItem/Article/SinglePrice
QTYI-1	/Order/LineItems/LineItem/Article/Amount
LEXC	/Order/LineItems/LineItem/Article/Price
LEXC*, VATP*	/Order/LineItems/LineItem/Article/Tax

* ソースデータは、ターゲットに書き込まれる前に関数により処理されます。
** これらのフィールドにより同じ型のフィールドのグループを介して反復が有効になります。

上のテーブルを参照として使用して、ソースコンポーネント内の上のアイテム全てから、ターゲットコンポーネント内の接続先へ、マッピング接続を描きます。この時点では、計算される値が必要なフィールドを省略します (すなわち、テーブル内で *のマークを持つフィールド)。これらは次のステップで処理されます。MapForce 内で接続を描くための一般的な情報に関しては、接続との作業を参照してください。

ヒント: MapForce コンポーネント内で名前別にフィールドを検索するには、「CTRL + F」を押します。

ステップ 2:変換 関数を追加する

IVDT アイテムの値を、小数点の値からxs:dateTime 型に変換するには、マッピングは、**core | conversion functions** ライブラリから**format-number** 関数を使用します。成功した例としては、変換中に先頭の 0 (ゼロ) を保持するを参照してください。

第 2番目の関数 **parse-date** の役割は、**format-number** により返されたYYMMDD 文字列の値をxs:dateTime 型に変換することです。2つの入力引数が存在します: (a) フォーマットされる値 (b) フォーマットマスク。フォーマットマスクは、基本的には、MapForce に、年、月、日付は、それぞれ2つの文字から構成されているように処理するように命令します。

上の関数をマッピングに追加するには、関数をライブラリウィンドウからマッピングエリアにドラッグします。関数に引数を提供するには、(メニューコマンド **挿入 | 定数**) を使用して 定数を入力します。MapForce 内の関数との作業に関しての情報は、関数と作業を参照してください。

ステップ 3:税率計算関数を追加する

式に従い税率を計算する関数は、**core | math functions** ライブラリ内で使用することのできる**multiply** と **round** です。また、関数は拡張可能です。(すなわち、引数の変数を取ります)。このサンプルでは、LEXC、VATP、整数 10を引数として取ります。最初の2つの値は、ソースTRADACOMS コンポーネント、から接続されており、定数により整数が与えられています。**multiply** 関数の結果は、**round** 関数を使用して最も近い整数に四捨五入されます。最後に、**round** 関数の結果は、2番目の**multiply** 関数を使用して、0.001により掛けられます。この結果は

アイテムXML コンポーネント (/Order/LineItems/LineItem/Article/Tax) の目的のアイテムに。に接続されます。

マッピングデザインの出力をプレビューするには、マッピング ウィンドウの下の部分にある **「出力」** タブをクリックします。

ステップ 4 (オプション) StyleVision でのスタイルマッピング出力

オプションで、ターゲットXML コンポーネントをStyleVision Power StyleSheet (.sps) ファイルにリンクすることができます。 (StyleVision, http://www.altova.com/stylevision.html を参照してください)。これにより、HTML、RTF、PDF、または 2007+ ファイルとして、マッピングの出力を生成することができます。 に関しての詳しい情報は、(StyleVision がインストールされている場合、各フォーマットの必要よ条件がそろっている場合、 StyleVision を使用してマッピングの出力のスタイルを指定するを参照してください)。任意で、ステップこのステップを行う場合、 XML コンポーネントのヘッダーをダブルクリックし、"StyleVision Power スタイルシートファイル"の横の、 MapForce Examples フォルダー内の**Order.sps** ファイルを参照します。マッピングエリアでは、HTML、PDF、RTF、および Word 2007+ タブをクリックし対応するフォーマットでマッピング出力を確認します。

7.5.11 UN/ EDIFACT

UN/EDIFACT はドキュメントインターチェンジのための事実上の金融業界の標準 (またUN 標準)です。MapForce は、D93.A - D.16A ディレクトリ内に含まれるメッセージへをサポートします。

MapForce 内のデフォルトの UN/EDIFACT バージョンは D.16A です。

D93A までさかのぼる全てのバージョンの個別のインストーラーが Altova Web サイトの MapForce コンポーネントページで使用することができます (http://www.altova.com/ja/components_mapforce.html)。

UN/EDIFACT に関しての詳しい情報は、 http://www.unece.org/trade/untdid/welcome.htm を参照してください。

7.5.11.1 UN/ EDIFACT 検証ルール

UN/EDIFACT ドキュメントの検証時には以下の項目がチェックされます :

- UNB ならびにUNZ セグメントが存在するか。
- UNB/S004 内に、妥当な日付 /時刻が指定されているか。
- UNB/0020 ならびにUNZ/0020 に同一の値が含まれるか。
- UNZ/0036 内に、 (それが存在する場合) 関数グループの数、またはメッセージ正しい数が含まれているか。関数グループが存在する場合その関数グループの数、またはインターチェンジ内に含まれているメッセージの数が記載されることになります。

各関数グループに対して、以下の項目がチェックされます :

- マッチするUNG とUNE のペアが含まれるか。
- UNG/S004 内に妥当な日付 /時刻の指定が含まれるか。
- UNE/0060 内に、関数グループ内に含まれているメッセージの正しい番号が含まれているか。

各メッセージに対して、以下の項目がチェックされます :

- マッチするUNH とUNT のペアが含まれるか。
- UNH/S009/0052 内に、含まれている関数グループのUNG/S008/0052 と同じ値が含まれているか。

- UNH/0062 ならびにUNT/0062 に同一の値が含まれているか。
- UNH/S009/0065 内に、正しいメッセージ型指定子が含まれているか。
- UNT/0074 内に、メッセージの中にあるセグメントの正しい数が含まれているか。

7.5.11.2 UN/ EDIFACT 自動データ補完ルール

UN/EDIFACT ファイルを生成する際、MapForce は、("自動データ保管 "または "自動補完 "ルールとして説明されている)ルールに従い、自動的に特定のフィールドまたは値を生成する可能性があります。この振る舞いを無効化するには、EDI コンポーネント設定内のチェックボックスの **不足しているフィールドの自動補完** のチェックを解除してください。

UN/EDIFACT の自動データ補完により、以下の項目が保証されます：

- UNB ならびにUNZ セグメントが存在する。
- UNG またはUNE の一報が存在する場合、もう一方のID も存在する。
- UNH ならびにUNT セグメントが存在する。
- UNB/S001 が存在する。その中にデータが含まれていない場合、ユーザーにより定義された設定から構文レベルと構文バージョンが使用されます。EDI コンポーネント設定の 「拡張 | 構文バージョン番号」 を参照ください。
- UNB/S002 ならびにUNB/S003 が存在する。
- UNB/S004 が存在する。データが含まれていない場合、現在使用されているEDI フォーマットの日付 /時刻が挿入されます。
- UNZ/0036 が存在する。データが含まれていない場合、関数グループまたはメッセージの数が計算され、挿入されます。
- UNZ/0020 の存在。データが含まれない場合、 UNB/0020 からの値がコピーされます。

メモ： このセクションにて言及されなかったフィールドが 自動データ補完により挿入または作成されることはありません。正しい値を自動的に確定することはできません。例えば、EDI コンポーネント内に ターゲット親要素のA と、それらの子アイテムx、y、z があり、そのうちy が必須アイテムとなっている場合、子要素の "y" がターゲットコンポーネントへマップされた時だけ、親要素のA が作成されます

関数グループのチェックにより、以下の項目が保証されます：

- UNG/0038 の存在。データが含まれない場合、メッセージ名が挿入されます。
- UNG/S006 ならびにUNG/S007 が存在する。
- UNG/S004 が存在する。データが含まれていない場合、現在使用されているEDI フォーマットの日付 /時刻が挿入されます。
- UNG/0051 が存在する。データが含まれていない場合、ユーザー定義設定から得られたコントロール代理 (controlling agency) にある最初の 2 文字が挿入されます。EDI コンポーネント設定 の 拡張 | コントロール代理」を参照ください。
- UNE/0060 が存在する。データが含まれていない場合、メッセージの数が計算され、挿入されます。
- UNE/0048 が存在する。値が含まれていない場合、UNG/0048 から得られた値がコピーされます。

メッセージのチェックにより、以下の項目が保証されます：

- UNH/S009/0065 の存在。データが含まれない場合、メッセージ名が挿入されます。
- UNH/S009/0052 とUNH/S009/0054 が存在する。
- UNH/S009/0051 の存在。が存在する。データが含まれていない場合、ユーザー定義設定から得られたコントロール代理 (controlling agency) にある最初の 2 文字が挿入されます。EDI コンポーネント設定 の 拡張 | コントロール代理」を参照ください。
- UNT/0074 の存在。データが含まれない場合、メッセージ内のにあるセグメントの数が計算され、挿入されます

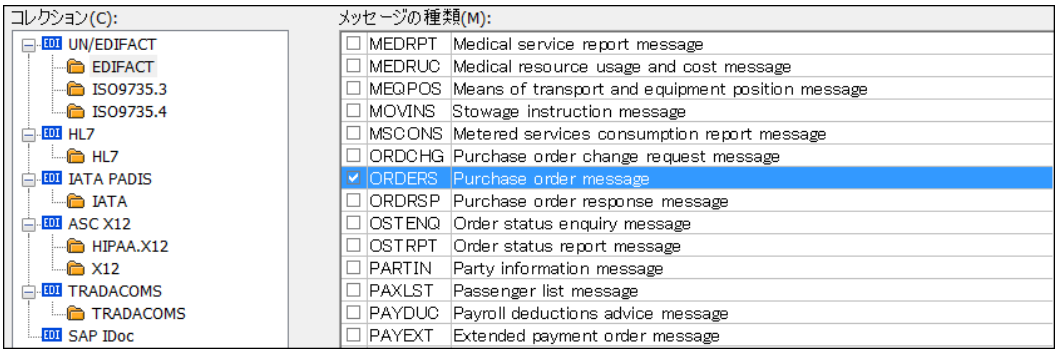
- UNT/0062 の存在。データが含まれない場合、 UNH/0062 からの値がコピーされます。
- UNH/0062 が存在する、データが含まれていない場合、UNT/0062 から得られた値がコピーされます。 (trailer セグメント番号だけがマッピングされている場合、ヘッダーセグメントの対応するフィールドが同じ値とともに与えられます)

7.5.11.3 サンプル UN/EDIFACT を XML にマッピングする

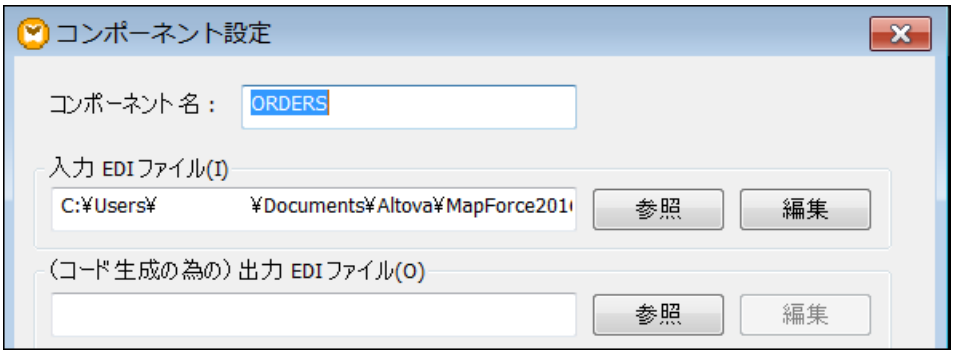
この演習では、UN/EDIFACT メッセージからのデータを XML スキーマ/ドキュメントへマッピングして、更なる処理を行うための XML ファイルを作成する方法について記述します。このサンプルで紹介されているマッピングは以下のパスで確認することができます :**<Documents>\ Altova\ MapForce2017\ MapForceExamples\ EDI_Order.mfd**.

ステップ 1 : UN/EDIFACT コンポーネントをマッピングに追加する

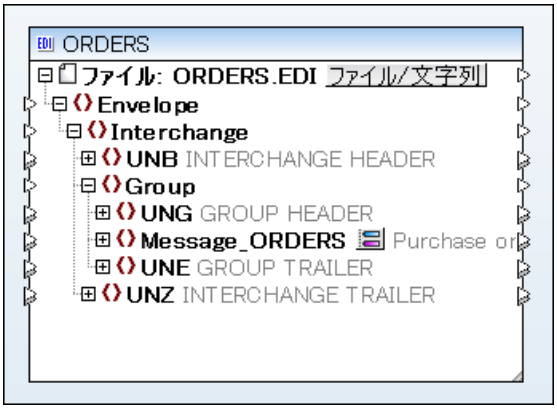
1. 新規マッピングを作成し、次の変換言語から1つ選択します :Java、C#、C++、または、BUILT-IN。このサンプルでは、が変換言語として使用されています。
2. 「**挿入**」メニューから、「**EDI**」をクリックします。
3. EDI のコレクションダイアログボックスから、EDIFACT を選択し、**ORDERS** を選択し、 「**OK**」 をクリックします。



4. サンプルEDI ファイルを提供するようにプロンプトされると、参照 をクリックして、file from the **<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリから **ORDERS.EDI** を選択し、開くをクリックします。コンポーネント設定ダイアログボックスが開かれ、入力ならびに出力 EDI インスタンスを選択することができます。これによりマッピングに追加する前に、EDI コンポーネントの設定を確認することができます。必要な場合これらの設定をいつでも変更することができます (次を参照 :EDI コンポーネント設定)。ORDERS.EDI ファイルが入力 EDI ファイルとして表示されることを確認してください。

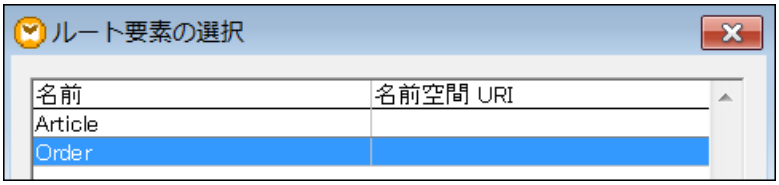


5. 「OK」をクリックします。EDI コンポーネントがマッピング領域に表示されます。Message_ORDERS ノードをダブルクリックして、子アイテムを確認します。コンポーネントウィンドウの右下をドラッグすることで、コンポーネントウィンドウの大きさを変更することができます。

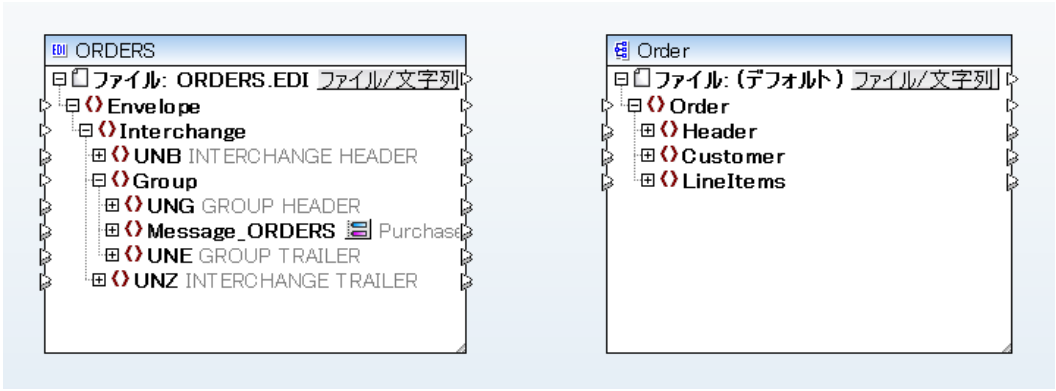


ステップ 2：ターゲットスキーマコンポーネントをマッピングに追加する

1. **「挿入」**メニューから XML スキーマ/ファイルをクリックして、**<Documents>\ Altova \ MapForce2017\ MapForceExamples** ディレクトリから**Order.xsd** を選択します。
2. サンプル XML ファイルを指定するよう促されるので、**「スキップ」**をクリックして、**Order** をターゲットドキュメントのルートとして選択します。



この時点で、ソース EDI コンポーネントとターゲット XML スキーマはマッピングエリアに表示されており、マッピング接続を開始することができます。



ステップ 3：EDI アイテムをマップする

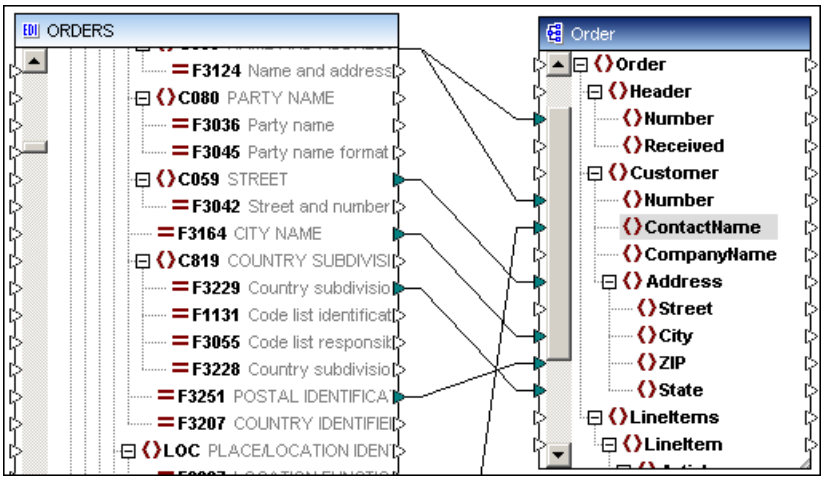
EDI コンポーネントには、選択したコレクション (ORDERS) から得られたメッセージの構造が表示されます。通常、全てのノードにデータが含まれることはなく、EDI を伴うプロジェクトを扱う際には、使用する EDI ドキュメントに十分精通している必要があります。

MapForce では、ソースアイテムとターゲットアイテム間に接続を描くことでマッピングを行うことができます。手順を追っての MapForce 内でのマッピング接続に関しては、接続と作業するを参照してください。

このサンプルでは、(**Group/Message_ORDERS** ノードから始まる)次のノードが最初にマップ(接続)されます:

ソース	ターゲット
BGM/C106/F1004	Order/Header/Number
SG2/NAD/C082/F3039	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

この時点では、マッピングは以下にあるようなものになるはずです:



マッピングプロセスを継続します:

ソース	ターゲット
SG29	Order/LineItems
SG29/LIN/C212/F7140	Order/LineItems/LineItem/Article/Number
SG29/IMD/C273/F7008	Order/LineItems/LineItem/Article/Name
SG29/QTY/C186/F6060	Order/LineItems/LineItem/Article/Amount
SG29/SG33/PRI/C509/F5118	Order/LineItems/LineItem/Article/SinglePrice

ステップ 4:日付のフォーマット

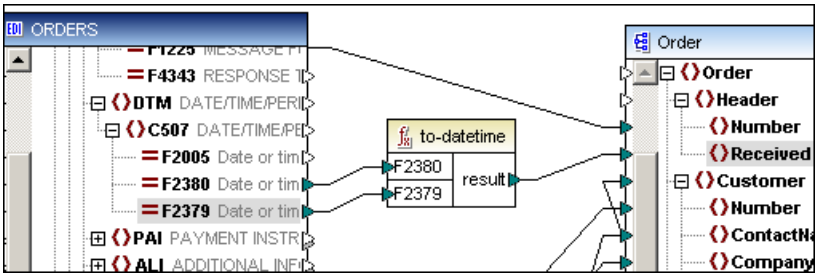
edifact ライブラリからto-datetime 関数をマッピング領域にドラッグします。内で関数を作業する方法に関しては、次を参照してください[関数と作業する]

DTM/C507 要素のF2380 とF2379 コンポーネントを適用することで、適正な方法でフォーマットされた受信側の

datetime を作成することができます。

ですから以下のフィールドをマップします：

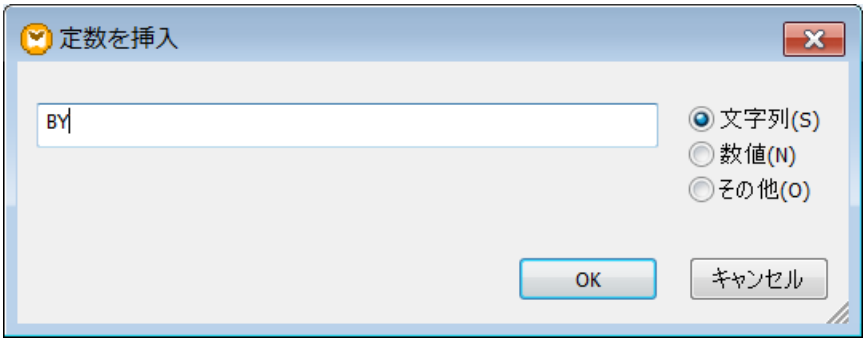
ソース	ターゲット
DTM/C507/F2380	**to-datetime** 関数の F2380 入力
DTM/C507/F2379	**to-datetime** 関数の F2379 入力
to-datetime 関数の**result** 出力	Order/Header/Received



ステップ 5：からの注文をフィルターする

ここで、"Buyer" からの購入注文をフィルタリングすることにします。フィルタリングはNAD (Name and Address) セグメントのパーティーを識別することで行われます。この場合、'BY' という値により "Buyer" 商品やサービスが販売される相手) が示されます。

1. **core** ライブラリからマッピング領域へ **equal** 関数をドラッグします。
2. マッピングにフィルターを追加します（**挿入**」メニューから、**「フィルター：ノード行」**をクリックします）。
3. マッピングに定数を追加します（**挿入**」メニューから、**定数**」をクリックします）。"BY" という値にテキストフィールドに "BY" を入力して定数を割り当てます：

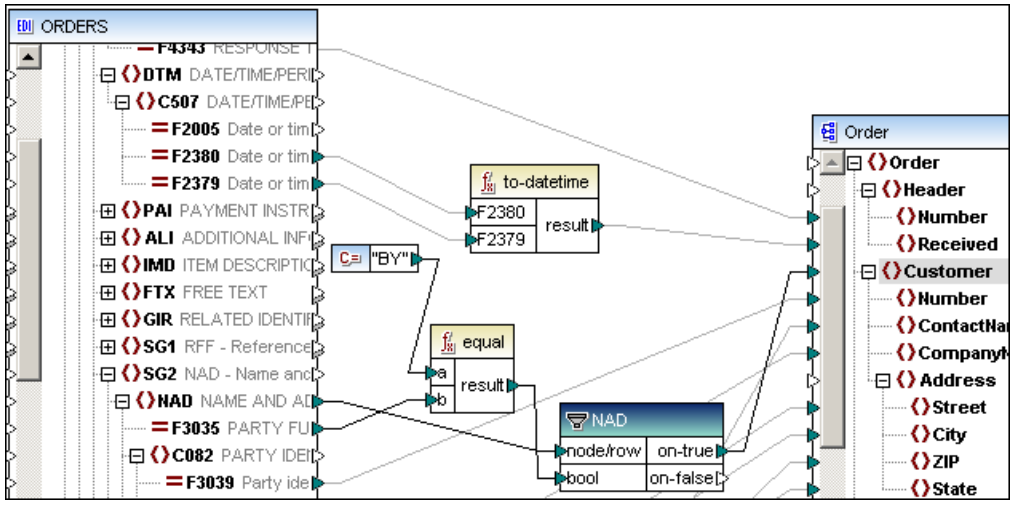


以下のアイテムをマップします：

ソース	ターゲット
SG2/NAD/F3035	**equal** 関数の**b** 入力
"BY" 定数	**equal** 関数の**a** 入力
equal 関数の結果	**filter** コンポーネントの**bool** 入力

ソース	ターゲット
SG2/NAD	フィルターコンポーネントの **node/row**
on-true の場合の**filter** コンポーネントの結果	スキーマの Order/Customer

この操作により、NAD ノードが、パーティーコードの "BY" により識別される "Buyer" を示している時だけ、データのマッピングが行われるようになります。



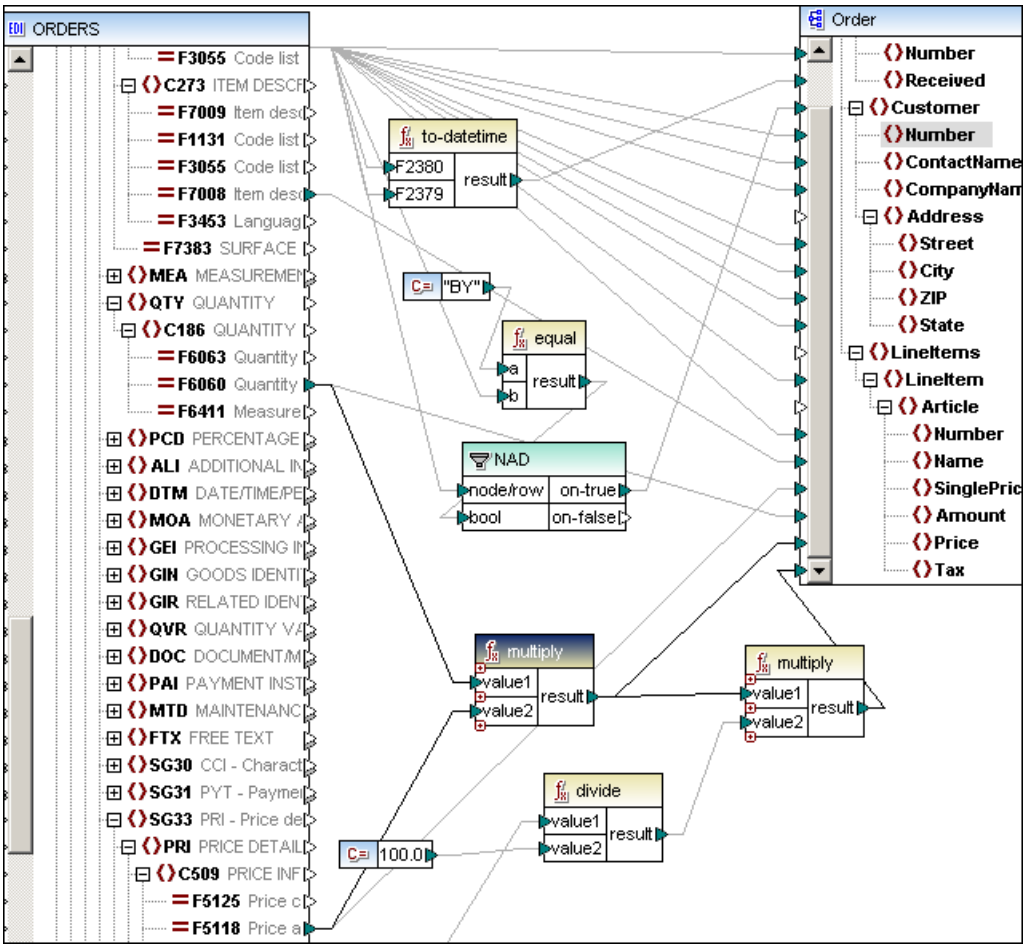
ステップ 6 : 価格と税金の計算

このタスクの最終的な仕上げとして、価格と税金の計算を行います。

1. core ライブラリから、**multiply** 関数を 2 つ、そして **divide** 関数を 1 つ、マッピング領域へドラッグします。
2. 定数コンポーネントを挿入します（**挿入 | 定数**）。数値が選択された状態で、"100.0" という値をテキストフィールドへ入力します。
3. 以下のアイテムをマップします :

ソース	ターゲット
SG29/QTY/C186/F6060	最初の**multiply** 関数の**value1**
SG29/SG33/PRI/C509/F5118	最初の**multiply** 関数の**value2**
最初の**multiply** 関数の結果	Order/LineItems/LineItem/Article/Price
SG29/SG40/TAX/C243/F5278	**divide** 関数の**value1**
定数 "100.0"	**divide** 関数の**value2**
最初の**multiply** 関数の結果	2番目の**multiply** 関数の**value1**
divide 関数の結果	2番目の**multiply** 関数の**value2**
2番目の**multiply** 関数の結果	Order/LineItems/LineItem/Article/Tax

マッピングは以下に示されているようなものになります：



出力タブをクリックすることで、変換が実行され、XML ドキュメントの変換結果が表示されます：

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <Number>ABC123456XYZ</Number>
    <Received>2004-04-30T17:42:00-09:00</Received>
  </Header>
  <Customer>
    <Number>123</Number>
    <ContactName>Michelle Butler</ContactName>
    <CompanyName>Nanonull, Inc.</CompanyName>
    <Address>
      <Street>119 Oakstreet Suite 4876</Street>
      <City>Vereno</City>
      <ZIP>29213</ZIP>
      <State>CA</State>
    </Address>
  </Customer>
  <LineItems>
    <LineItem>
      <Article>
        <Number>42</Number>
        <Name>Pizza Pepperoni</Name>
        <SinglePrice>7.2</SinglePrice>
        <Amount>1</Amount>
        <Price>7.2</Price>
        <Tax>0.648</Tax>
      </Article>
    </LineItem>
```

7.5.12 EDI 構造のカスタム化

デフォルトでは、公式の仕様に従い、MapForce は、マッピング上のEDI ファイルの構造を示します。多くの場合、デフォルトのEDI 構成を手動でカスタム化する必要はありません。しかしながら、多種の非標準 EDI フォーマットとバリエーションを処理する場合、カスタム化を必要とします。例えば、特定のデータ要素の構造または検証ルールを若干適用する場合があります。これは、カスタムEDI コレクションをビジネスのニーズに合わせ、MapForce に追加することにより達成することができる場合があります。

EDI バリエーション (カスタムEDI コレクション)をMapForce により既にサポートされる次の標準のために作成することができます :EDIFACT (ISO 9735)、ASC X12、HIPAA X12、HL7、または、TRADACOMS。新規のEDI 標準の始めからの作成はサポートされていません。

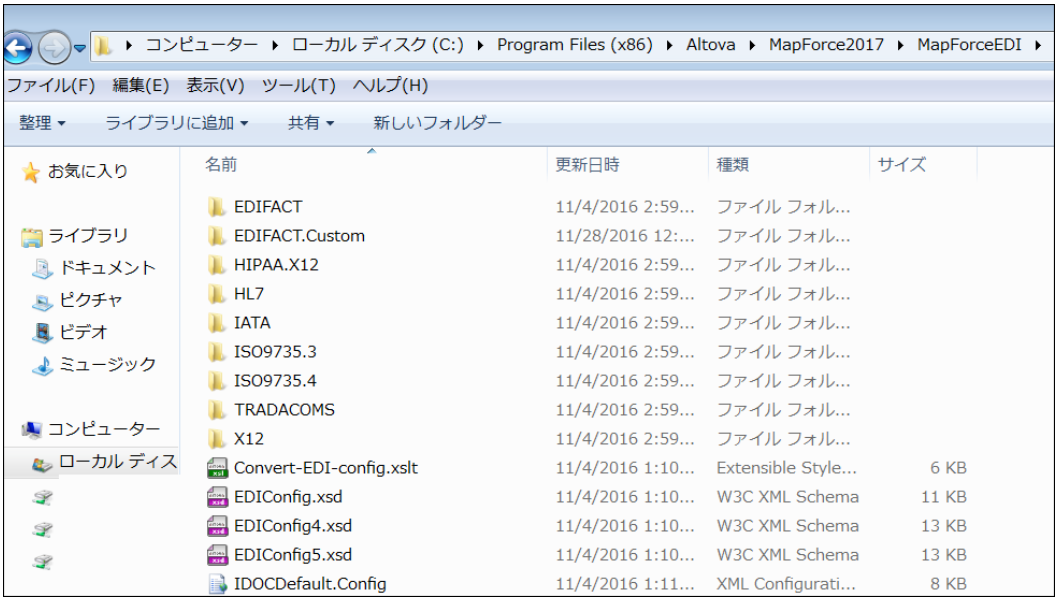
7.5.12.1 カスタム EDI コレクションの作成

"EDI コレクション"は、特定のEDI 標準に適用することのできるすべての構成ファイルを含むディレクトリです。デフォルトでは、内蔵のすべてのEDI コレクションはMapForce 内の次のパスで見つけることができます :**C:\Program Files \Altova\MapForce2017\MapForceEDI。**(32-ビットMapForce を 64-ビット オペレーティングシステムで作動する場合、ディレクトリパスを次のように調整します **C:\Program Files(x86)\...**)。

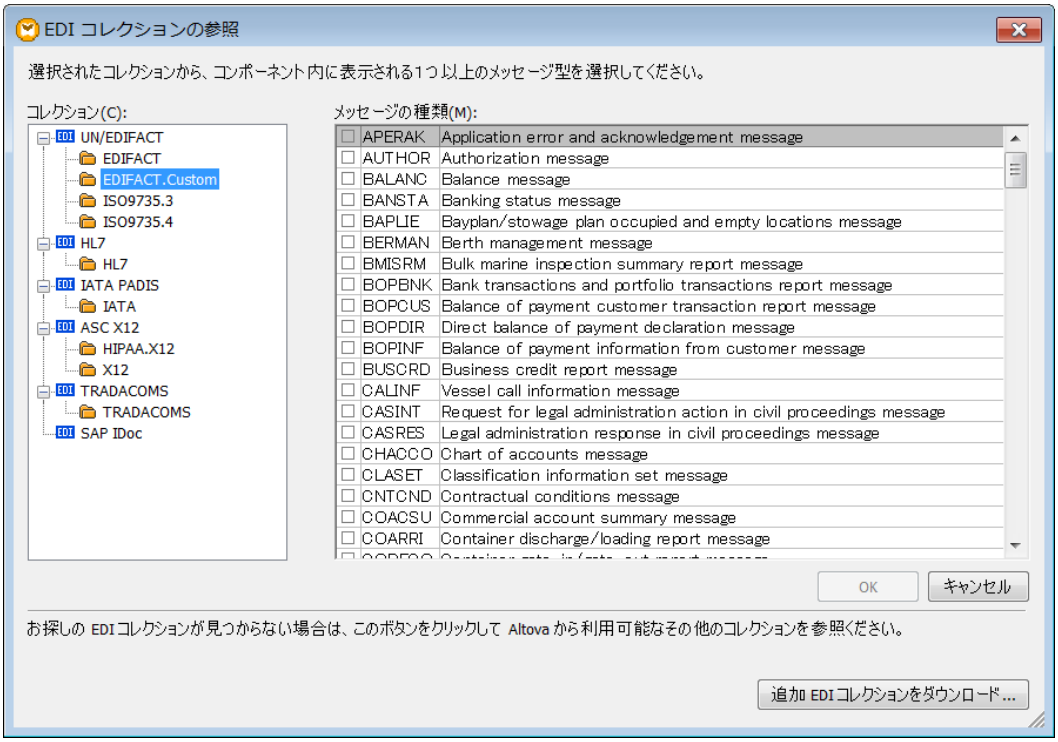
> ユーザーが責任を持ち、MapForce 内のEDI 検証および処理を上書きすることを希望する以外、**MapForceEDI** ディレクトリ内の元のファイルとフォルダーの内容を変更することは奨励されません。デフォルトの奨励されるアプローチは、既存のディレクトリを変更するよりも、新規のカスタムディレクトリを追加することです。このアプローチにより、新しいバージョンのMapForce にカスタムの構成を移行することがより簡単になります。

カスタム EDI コレクションの作成：

1. **C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ内に、カスタマイズする標準に対応するディレクトリの兄弟コピーを作成します（例："EDIFACT.Custom"）。



2. MapForce を開きます。
3. 「**挿入**」メニューから、「**EDI**」をクリックします。この段階では、カスタム EDI コレクションがデフォルトのエントリに加え「EDI コレクションの参照」ダイアログボックス内に表示されます。



メモ： C:\Program Files ディレクトリへの管理者の権利が必要な場合は必要に応じて変更してください。

作成されたカスタムコレクションは、まだカスタム化されていませんがデフォルトのものとは異なります。MapForce によりサポー

トされているデータフォーマットにカスタムEDI フォーマットからデータを両方向に変換するには、EDI コレクションをカスタム化する必要があります。この点については次で説明されています。現時点では、MapForce がEDI コレクションを認識できるように、ディレクトリが以下の必要条件を満たしていることが重要です：

1. **C:\Program Files\Altova\MapForce2017\MapForceEDI**). のサブディレクトリである必要があります。
2. 次のディレクトリうちの1つのコピーである必要があります：EDIFACT、HIPAA.X12、HL7、TRADACOMS、または、X12。

7.5.12.2 EDI 構成ファイル

EDI コレクション (ディレクトリ)には次の構成ファイルが含まれます：

ファイル	目的
EDI.Collection	**EDI.Collection** ファイルは実装中のカスタムEDI コレクション内のすべてのメッセージ型を定義します。MapForce 内の選択のために使用することのできるメッセージ型を追加、または、削除するために、このファイルを編集することができます (「EDI コレクションの参照」ダイアログボックス)。次を参照してください：メッセージ型の追加または削除.
Envelope.Config	**Envelope.config** ファイルは、MapForce 内のEDI マッピングコンポーネントの構造を定義します。(すなわち、**Envelope** シーケンスに囲まれた要素)。通常、このファイルを手動で編集してはなりません。
<Message>.Config	EDI コレクションディレクトリは、1つまたは複数の**<Message>.Config** ファイルを必要とします。このようなファイルは、EDI メッセージ型、および対応し、メッセージ内で使用されるグループとセグメント定義が含まれます。**<Message>.Config** ファイルは、**EDI.Collection** ファイルから参照されます。 特定のメッセージのフィールドに適応することのできる構造または検証ルールを変更する場合は、**<Message>.Config ファイル**を変更してください。このファイルに加えられる変更は、ローカルのカスタマイズと考えられます。(すなわち、これらは、特定のメッセージに適用されます)。例に関しては、次を参照してください EDIFACT メッセージのカスタム化 とANSI X12 トランザクションのカスタム化.
<Filename>.Segment	このファイルは、EDI ファイルのデータ要素、セグメント、複合、およびサブ複合を定義し、EDI ファイルを解析する際に使用します。 すべてのメッセージ内のフィールド (データ要素)に適用することのできる構造または検証ルールを変更する場合、このファイルに加えられる変更はグローバルにカスタム化されます。
<Filename>.Codelist	このファイルは、MapForce 内の入力と出力 EDI ファイルを検証するために、MapForce により使用される EDI コード値を定義します。通常、ここで定義されているコード値は、それぞれの EDI 標準による定義に対応します。このルールの例愛は、標準が外部コード値に頼り、可能な値の値を完全に強制できない場合です (次も参照：EDI コンポーネント検証)。 デフォルトの検証ルール を変更する場合、 codelist ファイルを変更することができます。例えば、検証を緩和、または、EDI コードリスト内で定義されていない新しいコードを追加する場合)。詳細に関しては、次を参照してください：フィールドの検証の弛緩 と外部ソースからコードの値を適用する。
ParserErrors.Config	このファイルは、X12 とHIPAA X12 標準に対してのみ使用すること

ファイル	目的
	ができます。MapForce 内の EDI コンポーネント上の "ParserErrors_Message" と "ParserErrors_Group" アイテムの構造を定義します。これらのアイテムは、ファイルの処理中の解析エラーの情報、および X12 997 (Functional Acknowledgement) と X12 999 (Implementation Acknowledgement) メッセージを作成するために使用することのできる情報を含んでいます。

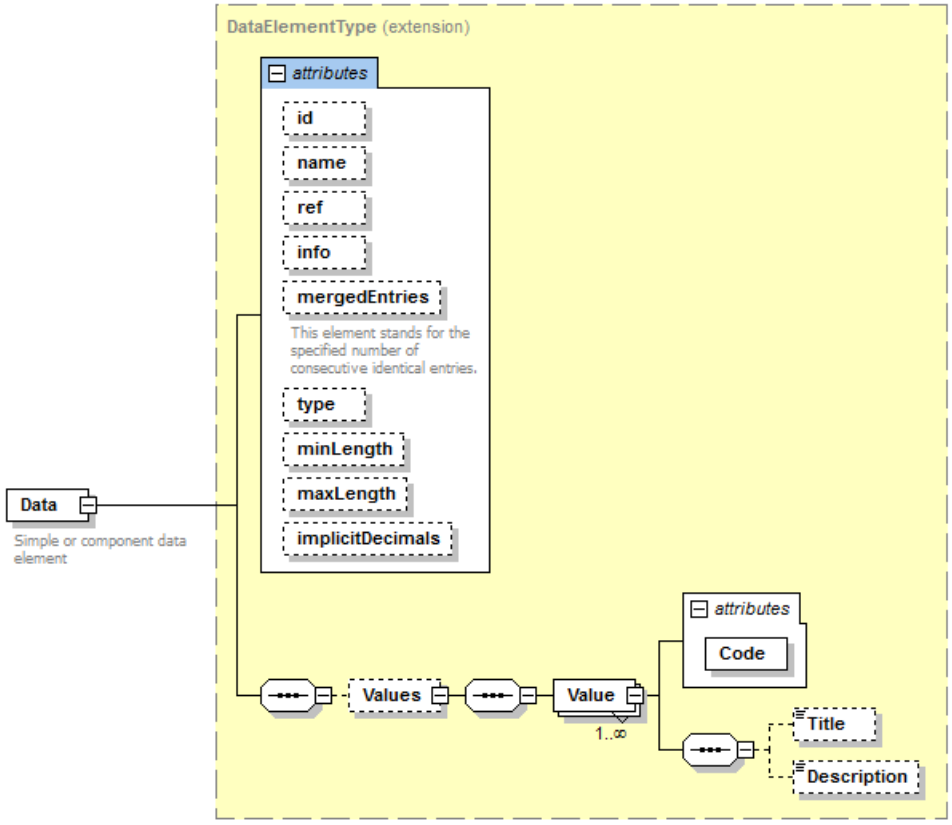
ファイルのすべては、XML ファイルで、XMLSpy などの XML エディターで編集することができます。**EDI.Collection** ファイルの例外を除いて、一部の XML スキーマが通常参照されます (次を参照 : EDI 構成スキーマ)。ファイルには、定義された明確なスキーマがありませんが、MapForce により内部で検証されます。

7.5.12.3 EDI 構成スキーマ

構成ファイルに変更する場合、使用することのできる XML スキーマに対して検証することは重要です。ディレクトリ **C:\Program Files\Altova\MapForce2017\MapForceEDI** で使用することのできるスキーマを確認することができます、以下のリストはリストの一部です :

ファイル	目的
EDIConfig.xsd (バージョン "3")	このスキーマは次の内蔵の MapForce EDI コレクションを検証します : • HL7 • X12 構成スキーマは、MapForce 2011 リリース 2 から有効です。バージョン 3 前のスキーマを使用する構成ファイルが存在する場合、このスキーマ (または、以降) を使用するために アップグレードされる必要があります。詳細に関しては、次を参照してください : 古い構成ファイルのアップグレード.
EDIConfig4.xsd	このスキーマは次の内蔵の MapForce EDI コレクションを検証します : • HIPAA X12 • TRADACOMS 構成スキーマは、MapForce 2011、リリース 3 から有効です。これには、X12 または HL7 フォーマットにデータをマップするために必要な可能性がある、カスタムの再帰的な階層構造を定義することのできる機能を含む、元の **EDIConfig.xsd** スキーマに加えられた複数の拡張が含まれています。
EDIConfig5.xsd	このスキーマは次の内蔵の MapForce EDI コレクションを検証します : • EDIFACT 構成スキーマは、MapForce 2017 から有効です。不完全な検証ルールを定義する機能が搭載されています。このスキーマ内では、文字レベルで高度なフィールドの検証ルールを定義することができます (次を参照 : 文字レベルの検証の構成)。

EDI 構成スキーマは、構成ファイル内に存在するすべての要素の構造に関する情報を提供します。例えば、下に表示される **EDIConfig4.xsd** スキーマ内に表示されるは EDI データ要素構造 以下のとおりです :



データ要素の構造 (XMLSpy のスキーマビュー)

7.5.12.4 古い構成ファイルのアップグレード

EDI 構成ファイルには、**MapForceEDI** ディレクトリ内に含まれるXML スキーマの1つへの参照が含まれています (次を参照 :EDI 構成スキーマ)。構成ファイルのスキーマとバージョンをビューするには、テキストエディターでファイルを開き、最初の行を確認してください:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd" Version="5">
```

MapForce 2017 は、MapForce 2011、リリース 3から搭載されている**EDIConfig.xsd** スキーマバージョン 3または、以降をベースにしたカスタムEDI 構成 を識別します。以前のバージョンと比較すると、このスキーマは同じEDI コンポーネント内の複数のメッセージ型から、データをマップすることができます。バージョン3以前のスキーマバージョンをポイントするカスタムEDI 構成ファイルを使用する場合、構成ファイルをアップグレードする必要があります。それ以外の場合、カスタムEDI コレクションをマッピングに追加する場合、次のメッセージが「EDI コレクションの参照」ダイアログボックスに表示されます:

> "選択されたEDI コレクションは、各コンポーネントに対し単一のメッセージ型をサポートする古い構成ファイルバージョンを使用しています。"

EDI 構成ファイルのスキーマを**EDIConfig.xsd** バージョン 2から**EDIConfig.xsd** バージョン 3 にアップグレードする方法が次に説明されています。

構成ファイルのアップグレード:

1. カスタム化されたEDI コレクション (例えば、 "EDIFACT.Custom") を含むフォルダーに**Envelope.Config** を元の構成フォルダー (例えば、 "EDIFACT") からコピーします。
2. **EDI.Collection**、を編集して、ルート要素の`Version` 属性を "2"から "3"に変更します。
3. `</Meta>` タグの後に**`<Root File="Envelope.Config"/>`** を追加します:

```
<?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. **ORDERS.Config**、を編集して、ルート要素の`Version` 属性を "2"から "3"に変更します。
5. **`<Format standard="EDIFACT"/>`** (、または、場合によっては、X12、または、HL7) を`</Meta>` タグの後に追加します。
6. **`<Group name="Message"...>`** を**`<Group name="Message_ORDERS"...>`** (または、カスタムメッセージ型)に名前を変更し、外部のグループレベルを削除します (Envelope"、"Interchange" およびそのセグメント)は以下のとおりです:

```
<?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>
        <Segment ref=" UNZ"  minOccurs=" 0" / >
      </ Group>
    </ Group>
   </Message>
</Config>
```

構成ファイルを編集中にマッピング開かれている場合、再ロードされる必要があります。 "Message" to "Message_ORDERS" アイテムから自動的に接続か再マップされます。

7.5.12.5 メッセージ型の追加または削除

カスタムEDI メッセージ型 (または、X12 用語内のトランザクション型) を次の方法で追加することができます :

1. カスタムEDI コレクションが保管されているディレクトリ内に、コレクションに追加されるメッセージ (トランザクション)の各型のために**<Message>.Config** ファイルを作成します。
2. **EDI.Collection** ファイルを、使用することのできるメッセージ (トランザクション)型への参照を作成するために変更します。

既存のメッセージ型を削除する :

- **EDI.Collection** ファイル内の必要とされるMessage 要素 のコメントアウト。

<Message>.Config ファイルの追加および構成

作成する各 **<Message>.config** ファイルは、構成スキーマ (次を参照 :EDI 構成ファイル) に対し妥当である必要があります。問題を回避するには、各メッセージ構成ファイルの本文内の次の要素に注目してください :

- Meta
- Format
- Include
- Message

`Meta` 要素は、サンプルのために、仕様バージョン、リリース、機関を表示します :

```
<Meta>
   <Version>D</Version>
   <Release>14B</Release>
   <Agency>UN</Agency>
</Meta>
```

`Format` 要素は、ベース標準の名前を示します、例 :

```
<Format standard="EDIFACT"/>
```

以下の値は妥当です :

- Fixed
- EDIFACT
- X12
- HL7
- TRADACOMS (この値は、**EDIConfig4.xsd** スキーマまたは、以降、を必要とします、次を参照してください EDI 構成ファイルとスキーマ)

<Message>.Config ファイル内の `Include` 要素は、追加構成ファイルへの参照を指定します。例えば、下のコードリストは2つの segment ファイルと1つの codelist ファイルへの参照を含んでいます。ファイルは、すべてのセグメント、複合、およびデータ要素の定義をグローバルなレベルで保管します (これは、各メッセージ内でこれらが複数回発生することを回避するためです)。同様に、 codelist ファイルは、グローバルなレベルで、各フィールドに適用することのできる検証ルールを定義します。

```
<Include href="Admin.Segment"/>
<Include href="EDSD.Segment"/>
<Include href="UNCL.Codelist"/>
```

`Include` 要素による参照は、**<Message>.Config** ファイルで同じディレクトリで存在する必要があります。

各 **<Message>.Config** ファイル内の `Message` 要素は、メッセージの構造 (すなわち、グループ、セグメント、およびデータ要素の階層構造) 、および検証のために使用されるコードの値を (適用することができる場合) 定義します。メッセージの構造に関する情報は、次を参照してください メッセージの構成の変更。既存の メッセージの変更のサンプルに関しては、次のトピックを参照してください :

- EDIFACT メッセージのカスタム化
- ANSI X12 トランザクションのカスタム化
- メッセージレベル (インライン) でコードの値を定義する

EDI.Collection ファイルの変更

カスタム **<Message>.Config** ファイルを追加すると、`Message` 要素が **EDI.Collection** ファイルに追加される必要があります。各 `Message` 要素には次の属性が存在します：

型	メッセージ型の公式なコードを指定します (例："BALANC", "999").
ファイル	この特定のメッセージの型の仕様をるくむファイル名を指定します。実際のファイルは同じディレクトリに存在する必要があります。
説明	MapForce の視覚的なユーザーインターフェイスに表示されるこのメッセージの型の説明を提供します。

妥当な **EDI.Collection** ファイルのサンプルに関しては、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ内の MapForce 内蔵の EDI コレクションを参照してください

7.5.12.6 メッセージの構成の変更

各 **<Message>.config** ファイル内では、`Message` 要素の構造がカスタム化される EDI 標準の XML スキーマにより定義されています。このスキーマは、**EDIConfig.xsd** (または、類似した名前で呼ばれます) 次を参照してください EDI 構成スキーマ。メッセージの構造を構築、または、変更するには、カスタム EDI 標準と XML スキーマ。のドキュメントを確認する必要があります。

セグメント、データ要素、複合の追加方法

データ要素、セグメント、複合、およびサブ複合を追加するには、`<Data>`、`<Segment>`、`<Composite>` と `<SubComposite>` 要素 を以下のように作成します：

- .segment ファイルには、再利用できる、グローバルに定義されたセグメント、とデータ要素のリストが含まれている必要があります。
- .config ファイルは、各 メッセージ (または、トランザクション) 型に対応しており、そのメッセージ (セグメントの階層構造、複合、および、データ要素) の実際の構造を含んでいる必要があります。

データ要素 を (segment ファイル内で) 1度定義することができるようになり、(config ファイル内の) 複数のメッセージ内で再利用することができます。

データ要素を参照するには、`ref` 属性を使用します。例えば、下のコードリストは、データ要素 F100 が HIPAA.X12 コレクションの **X12.Segment** ファイル内で定義されていることを示しています。

```
<Data name="F100" type="string" maxLength="3" minLength="3" info="Currency
Code" />
```

X12.segment ファイル

すべてのセグメントでは、このデータ要素を必要とするすべてのトランザクション型内で、このデータ要素を参照できます。下のコードリストは、"Health Care: Professional" (837P) トランザクションの "CUR" セグメントが前に定義された F100 データ要素を参照していることを示しています：

```
<Include href="X12.Segment" />

...

<Segment name="CUR" minOccurs="0" info="Foreign Currency Information">
    <Data ref="F100" info="Currency Code" />
</Segment>
```

837P.Config ファイル (X12)

参照は、同じファイル、または、外部 ファイルに存在することができます。上のコードリストでは、 config ファイルには外部 .segment ファイルが含まれています。このため、`<Include href="X12.Segment" />` ステートメントが、外部 **X12.Segment** ファイル。に参照を加えるために、**837P.Config** ファイルの最初に存在します。

カスタムセグメントの追加方法を説明する他のサンプルに関しては、次を参照してください サンプル :TA1 セグメントの追加.

グループとループの作成方法

EDI 構造を作成し、変更するには、親ノード (グループ)の下のセグメントまたはデータ要素をグループ化することを必要とします。このメッセージを構成する複数のセグメントをグループ化するため、EDI コンポーネント上のメッセージもグループです。グループは、ネストされている可能性があり、(子グループを持つこともできます)。また、複数回繰り返される可能性もあります (この場合は、ループになります)。グループとループは、`<Group>` 要素を追加することで作成することができます。`maxOccurs` 属性を使用して、ループの発生回数を設定します。例えば、次のコードリストは、ループが25回まで任意で発生でき、2つのセグメントにより構成されていることを示していますることを:

```
<Group name="LoopSAC" maxOccurs="25" minOccurs="0">
   <Segment ref="SAC"/>
   <Segment ref="CUR" minOccurs="0"/>
</Group>
```

850.Config ファイル (X12)

データ要素の型とサイズの指定する方法

データ要素またはセグメントの`type` 属性を使用して、(文字列、小数など)型を指定します。可能なデータ型 (`type` 属性)は、通常デフォルトで搭載されている構成ファイル内で使用されているXML スキーマ型であることができます。他の XML スキーマ単純型は、一時的に使用することができますが、保証はされません。次の単純型はサポートされていません: "anyType"、"ENTITIES" と "QName"。

データ要素の`minLength` と`maxLength` 属性を使用して、最小値 と最大値を指定するために使用します。

必須と任意のデータ要素の指定方法

データ要素が必須であることを指定するには、`minOccurs` 属性の値を設定しないでください (この属性には、デフォルトとして "1"を持ちます。これはフィールドがデフォルトで必須であることを意味します)。データ要素が任意であることを指定するには、`minOccurs` 属性を "0"に設定します。例えば、下のコードリストは 2つの必須のデータ要素 (F1331、F1332) と1つの任意のデータ要素 (F1325) を定義しています。

```
<Composite name="C023" info="Health Care Service Location Information">
   <Data ref="F1331"/>
   <Data ref="F1332"/>
   <Data ref="F1325" minOccurs="0"/>
</Composite>
```

7.5.12.7 複数の連続したデータ要素の扱い

EDI セグメントは、同じ名前の複数の連続した要素の使用を許可します。例えば、X12 標準の "850 - PURCHASE ORDER" トランザクションセットの N2 セグメントに適用することができます。仕様に従うと、N2 セグメントは、(1つの必須、1つの任意) 2つの連続した英数字 "Name" フィールドを許可します :

REF	ELE ID	NAME	RPT	ATTRIBUTES
01	93	Name	M	AN 1/60
02	93	Name	O	AN 1/60

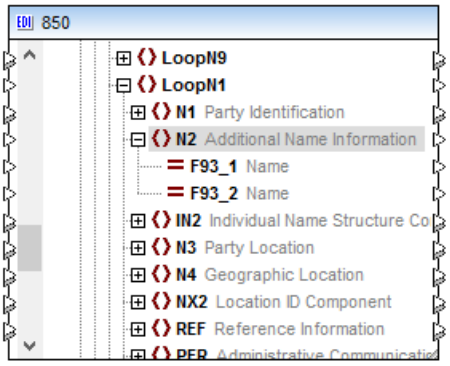
EDI 構成を作成、または、変更するために、データ要素を扱うには以下の2つの可能性があります。

アプローチ 1 : 各連続したデータ要素のための個別の マッピング ノードを作成する

このアプローチでは、セグメント ファイル内のデータ要素の複数の発生を作成します。この場合、新規のノードは、データ要素の各発生のためにマッピング上に表示されます。この構成を考慮してください :

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" nodeName="F93_1"/>
   <Data ref="F93" minOccurs="0" nodeName="F93_2"/>
</Segment>
```

マッピング上では、この構成は、2つの個別の F93 ノード , を作成します。それぞれは、F93 の発生に対応しています :



個別の F93 ノードを持つ X12 EDI コンポーネント

このアプローチの使用は、データ要素の発生が意味を成す場合役にたちます。このため、マッピング上に個別のノードを表示するためにも役に立ちます。

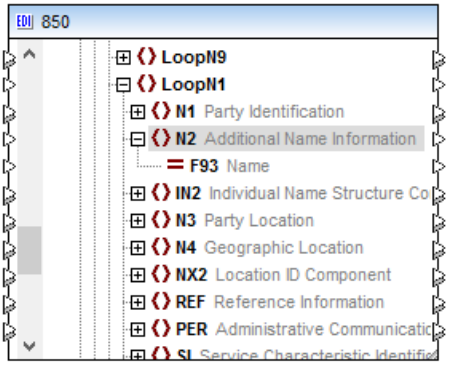
アプローチ 2 : データ要素のすべての発生のために単一ノードを作成する

このアプローチでは、データ要素のすべての発生を単一のノードとして扱います。これを行うには、属性を対応するデータ要素に追加します。下のコードリスト内は、データ要素 F93 の `mergedEntries` 属性 は "2" に設定されています。EDI インスタンス ファイル内のこのフィールドの 2つの連続した発生を許可します。

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
```

```
</Segment>
```

マッピング上では、F93 データ要素に対応するノードは、単一ノードとして表示されます：



個別のF93 ノードを持つX12 EDI コンポーネント

ソースEDI ファイルにこの要素の(空ではない)複数の発生が存在する場合、エントリをマージしたデータ要素からのデータのマッピングは、ターゲット側に複数の複数要素を作成します。例えば、N2 セグメントが2つの連続した発生を含む箇所のソースEDI ファイルを想定します。最初の発生は、"Michelle Butler" で、2番目の発生は、"Mrs" です。

```
N2+Michelle Butler+Mrs+'
```

このようなEDI ファイルからのデータのマッピングは、ターゲット側に次の出力を作成します (<ContactName> の発生に注意してください)：

```
<Customer>
      <Number>123</Number>
      <ContactName>Michelle Butler</ContactName>
      <ContactName>Mrs</ContactName>
      <CompanyName>Nanonull, Inc.</CompanyName>
</Customer>
```

データ要素の各発生が意味をなさない場合、また、各発生のために、マッピングに表示するためにノードを分割することを希望しない場合、このアプローチの使用は適切です。例えば、複数の発生を持つデータ要素は、住所を構成する複数のラインエントリを保管することができます。このような場合、"マージされたエントリ"のアプローチは、各住所のラインのために個別のノードを使用するよりも役に立つ可能性があります。

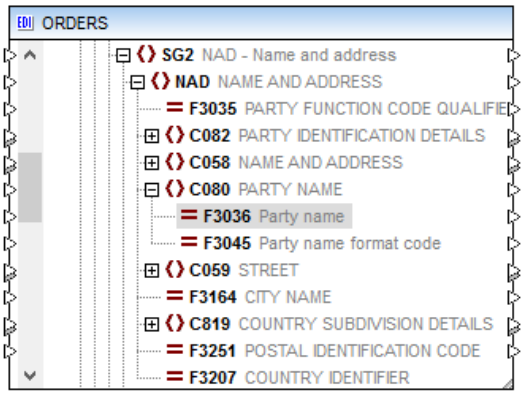
MapForce によりデフォルトで与えられる構成ファイルの一部は、マージされたエントリを使用する場合があり、このため、データ要素の複数の連続した発生のためのマッピング上で1つのノードのみが表示されます。データ要素の各発生のために、個別のノードが必要な場合は、EDI 構成をカスタム化して行ってください。例えば、次を参照してください マージされたエントリを異なるノードに分割する。

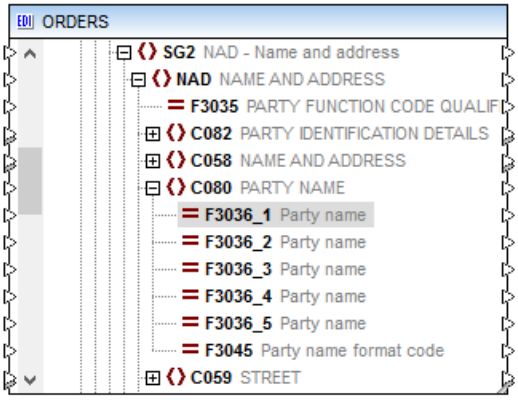
7.5.12.8 マージされたエントリを異なるノードに分割する

MapForce によりデフォルトで与えられるEDI 構成ファイル内のデータ要素の一部は、ソースEDI ファイル内で対応するデータ要素が複数回発生する可能性があるにも関わらず、マッピング上で1つのノードのみを表示するように構成されている場合があります (次も参照：複数の連続したデータ要素の扱い)。このようなデータ要素の一例は、UN/EDIFACT 標準のF3036 です。仕様に従うと、F3036 は、("Party name") 複合内の C080 内で連続して5回まで発生することができます (次を参照：http://www.unece.org/fileadmin/DAM/trade/untdid/d15b/trcd/trcdc080.htm)。

MapForce 内のデフォルトのUN/EDIFACT コレクションでは、データ要素 F3036 は、マッピング上で単一のノードで表示されています。マッピング接続をこのノードからターゲットノードに描くと、ソース内のF3036 の発生が存在するため、接続はターゲット内に制限なくアイテムを作成することができます。 (これは、ソース内の各アイテムのために、ターゲット内に1つのアイテムが作成されるMapForce 接続の基本ルールに従っています。)。しかしながら、これが希望される結果ではない場合、データ要素の各発生のために個別のノードが必要とされます。

この例の目的は、ですから、F3036 の各発生のための個別のマッピングノードを表示するカスタムUN/EDIFACT 構成を作成することです。下のイメージは、カスタム化の前後のEDIFACT マッピングコンポーネントを表示しています。





カスタム化の前 *カスタム化の後*

F3036 データ要素を複数のマッピングノードに以下のように分割します :

1. 既存のUN/EDIFACT コレクションを開始点として使用し、カスタムEDI コレクションを作成します (次を参照 :カスタムEDI コレクションの作成)。これにより、変更は、EDI 構成には、影響をおよぼさないことを保証します。
2. XML エディター内でカスタム コレクションの**EDSD.Segment** ファイルを開きます。(EDI フレーバーをカスタム化する場合、特定のEDI フレーバーの対応する segment ファイルを変更します。例えば、X12 のためには、**X12.Segment** ファイルになります。)
3. 複合 C080 を検索し、F3036 データ要素の`mergedEntries="5"` 属性を削除します。

```
<Composite name="C080" info="PARTY NAME">
   <Data ref="F3036" mergedEntries="5"/>
   <Data ref="F3045" minOccurs="0"/>
</Composite>
```

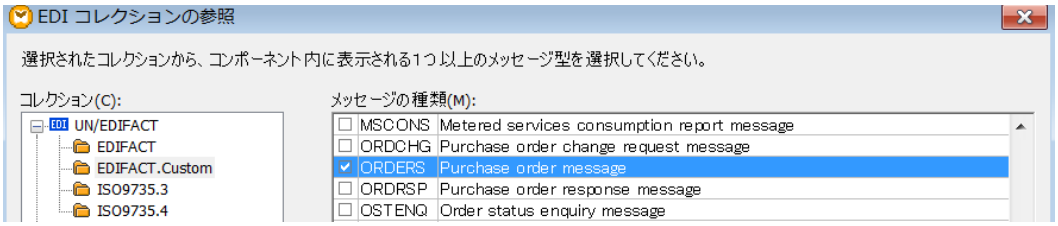
4. `minOccurs="0"` 属性を、最初のフィールド以外のすべてのフィールドに追加します。理由は、UN/EDIFACT 標準によると、フィールドの最初の発生のみが必須であるからです。その他は任意です。
5. 各フィールドのために一意のノード名を入力します (例えば、F3036_1、F3036_2、など)。この時点では、構成は以下のようになります :

```
<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>
```

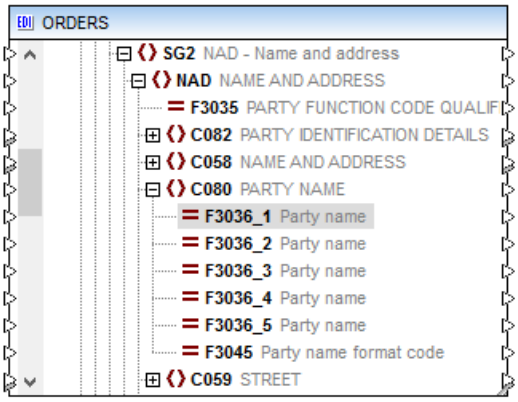
6. **EDSD.Segment** ファイルを保存して編集します。

MapForce 内で変更されたEDI コレクションを以下の方法で確認、および作業することができます :

1. **挿入**」メニューから、「**EDI**」をクリックします。
2. プロンプトされると、作成済みのカスタムコレクションを選択します (このサンプルでは、 "EDIFACT.Custom")。
3. C080 セグメントを使用するメッセージ型を選択してください (このサンプルでは、 "ORDERS - Purchase Order Message")。



変更されたEDI コンポーネントがMapForce 内に表示されます (F3036 データ要素の各発生には、マッピング上で個別のノードが存在します)。



EDI フレーバーに関係なく、このサンプル内で説明されているこのテクニックは、他のデータ要素のために マージされたエントリを分割するために使用することができます。

7.5.12.9 データ要素のコードの値の変更の可能性

データ要素の可能の値は構成ファイルを使用して変更することができます。他の構成情報同様、メッセージレベル (ローカルでのカスタム化) または、グローバル (特定のデータ要素を使用する可能性のあるすべてのメッセージで) で可能なコード値を構成することができます。

> 他のカスタムの構成のように、コード値をカスタマイズするには、デフォルトのMapForce 構成を変更する代わりに、カスタムEDI コレクションを作成し、すべてのカスタム化を適用することが奨励されます (次を参照 : カスタムEDI コレクションの作成)。

使用することのできるカスタム化オプションは、カスタムEDI コレクションにより使用されるスキーマバージョンにより異なります (次を参照 : EDI 構成スキーマ)。つまり、新しいスキーマに対して更に高度な構成オプションを使用することができるのです。特に、各スキーマはカスタムコードを以下のようにサポートします :

- **EDIConfig.xsd**、を使用する場合、グローバルなレベルでカスタムコードを追加するために既存の codelist ファイルを変更することができます。
- **EDIConfig4.xsd**、を使用する場合、カスタムコード値をグローバルなレベルだけではなく、メッセージレベルで追加することができます。

- **EDIConfig5.xsd**、を使用する場合、高度な構成ルールを文字レベルで定義する機能を含む、上記のすべてのオプションを使用することができます。(次を参照：文字レベルの検証の構成)。また、データ要素には不完全な検証が存在すると宣言することができます (次を参照：フィールドの検証の弛緩)。

グローバルなカスタム化

データ要素のためにグローバルにカスタムコード値を追加するには、カスタム EDI コレクションの codelist ファイルを編集します。最初に、値を変更するデータ要素 (フィールド) 検索し、新しい<Value> 要素として必要とされるカスタム値をすべて以下に示されるように追加します：

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="ISO6346.Codelist"/>
   <Codelist>
      <Field Name="F1001" Length="3" Type="Alphanumeric">
         <Title>Document name code</Title>
         <Description>Code specifying the document name.</Description>
         <Values>
            <Value Code="1">
               <Title>Certificate of analysis</Title>
               <Description>Certificate providing the values of an
analysis.</Description>
            </Value>

>>> add your custom values here...

        </Values>
      </Field>
   </Codelist>
</Config>
```

UNCL.Codelist ファイル (UN/EDIFACT コレクション)

上のコードリストに示されているように、各 <Value> 要素には、このデータ要素のために期待される実際の値コードを指定する Code 属性が存在します。 <Title> と<Description> 要素は、この値コードに関する詳細の情報を提供します。

上のコードリストには、MapForce に **UNCL.Codelist** ファイルを同じスキーマを使用する。。外部ファイル (**ISO6346.Codelist**) 内で定義されている追加コード値を検索するように命令する Include ステートメントが存在します。

例えば、次を参照してください 外部ソースからコードの値を適用する。

ローカルなカスタム化

メッセージレベルでカスタムコード値を追加するには、メッセージ型に対応した config ファイルを編集する必要があります。詳細に関しては、次を参照してください：メッセージレベル (インライン) でコードの値を定義する。

7.5.12.10 サンプル :TA1 セグメントの追加

TA1 セグメントは、任意のセグメントで、インターチェンジのレセプション、および、エンベロープセグメントの文法的な正確性を確認するために使用されます。構成ファイルを変更することにより、セグメントを MapForce 内の EDI X12 と EDI

HIPAA コンポーネントに追加することができます。

EDI コンポーネント内にTA1 セグメントを含むト:

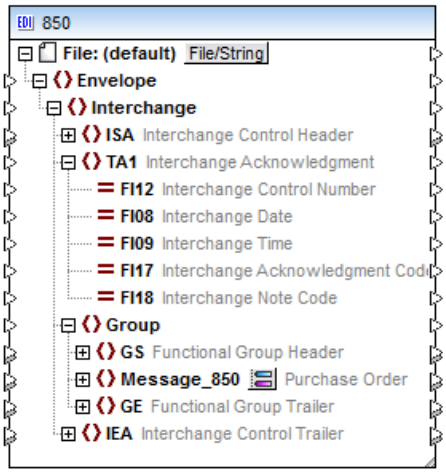
1. HIPAAA.X12 コレクションをモデルにして、カスタムEDI コレクションを作成します (次を参照 :カスタムEDI コレクションの作成)。これは、変更がデフォルトの構成に影響を与えないことを保証します。
2. カスタムEDI コレクションの**Envelope.config** ファイルをXML エディター内で開きます。
3. ISA とGroup セグメントの間のテキスト**** を解析します。そしてファイルを保存します。これは以下のようになります :

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig.xsd" Version="3">
   <Meta>
      <Release>6020</Release>
      <Agency>X12</Agency>
   </Meta>
   <Format standard="X12"/>
   <Include href="X12.Segment"/>
   <Include href="X12.Codelist"/>
   <Include collection="EDI.Collection"/>
   <Group name="Envelope">
      <Group name="Interchange" maxOccurs="unbounded">
         <Segment ref="ISA" minOccurs="0"/>
         <Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>
         <Group name="Group" maxOccurs="unbounded">
            ...
         </Group>
         <Segment ref="IEA" minOccurs="0"/>
      </Group>
   </Group>
</Config>
```

上でハイライトされたラインは、**X12.Segment** ファイル内で既に定義されているTA1 セグメントに参照を追加します。MapForce 内でTA1 セグメントの構造を確認することができます :

1. **「挿入」**メニューから、**「EDI」**をクリックします。
2. 「EDI コレクションの参照」ダイアログボックスから、作成されたカスタムコレクションとメッセージ型を選択します (例えば、"850 -Purchase Order")。

カスタム化されたコンポーネントには、TA1 セグメントが追加されています。必要に応じてこのカスタム化されたコンポーネントにマッピングの接続を描くことができます。



任意のTA1 セグメントが追加されたカスタム化されたX12 EDI コンポーネント

7.5.12.11 例 :EDIFACT メッセージのカスタム化

この例は、EDIFACT メッセージをカスタム化し、MapForce が非標準、または、変更されたEDIFACT フォーマットを処理する方法について説明しています。ORDERS メッセージの若干拡張されたCTA (Contact Information) セグメントを使用して、カスタムEDIFACT ファイルを処理するサンプルのマッピングは次のパスで見つけることができます : **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Orders-Custom-EDI.mfd**。このマッピングを次のカスタム化を行う前に開いた場合、MapForce は、正確に検証できないことに注意してください。この理由は、マッピングが、使用中のマシン上にまだ存在しないカスタムEDIFACT 構成を参照しているからです (この例の中でこの構成は作成されます)。

Orders-Custom.EDI ファイルからデータを読み取るサンプルのマッピングは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリ内にあります。このファイルをテキストエディターで開くと、カスタム化されたCTA セグメントが使用されていることに気がつきます。具体的には、9列目では、**「Mr」**エントリを含みライン 11列目では、**「Mrs」**エントリを含みます :

```
 1      UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
        ADDR+970101:1230+00000000000001++ORDERS++++1'
 2      UNH+0001+ORDERS:S:93A:UN'
 3      BGM+221+ABC123456XYZ+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'
```

上記のようなカスタムEDI ファイルからデータをマップする前に、カスタムEDIFACT 構成が作成される必要があります。この例では、個人のタイトルに対応するようにCTA セグメントに新規フィールドを持つように拡張するために、デフォルトのEDIFACT 構成をカスタマイズします。参照のために、カスタマイズの最終結果は、ZIP アーカイブ (**EDIFACT.Nanonull.zip**) として **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリ内にあります。下の手順は、ZIP ファイルが **..\Program Files\MapForceEDI** ディレクトリ

に既にアンパック済みであると想定しています。

カスタマイズファイルのセットアップ

1. MapForce インストールディレクトリ (**C:\Program Files\Altova\MapForce2017**) を開きます。 32-ビット MapForce が 64-ビット オペレーティングシステム上で作動している場合は、パスを**C:\Program Files (x86)\Altova\MapForce2017\MapForceEDI** に調整してください。
2. **..\MapForceEDI** ディレクトリの下に、**EDIFACT.Nanonull** という名前のディレクトリを作成します。
3. …**MapForceEDI\EDIFACT** フォルダーから次のファイルを**EDIFACT.Nanonull** フォルダーにコピーします。

   - **EDI.Collection**
   - **Envelope.Config**
   - **UNCL.Codelist**
   - **Admin.Segment**
   - **EDSD.Segment (Electronic Data Segment Definition)**
   - **ORDERS.Config**

   各ファイルの目的に関する詳細に関しては、次を参照してください EDI 構成ファイル.

メモ: ファイルの属性を編集できるように **「読み取り書き込み」** に変更してください。

EDI.Collection ファイルを構成する。

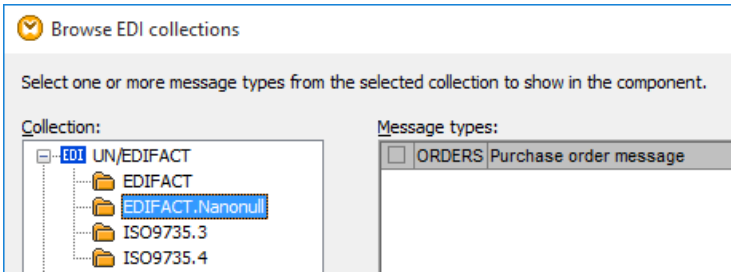
1. XMLSpy または エディターで**EDI.Collection** ファイルを開きます。
2. "ORDERS" メッセージ以外のすべての "Message" 要素を削除します。 <Messages> タグが保持されていることを確認してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Version>D</Version>
      <Release>16A</Release>
      <Agency>UN</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase
order message"/>
</Messages>
```

3. ファイルを保存します。

新規の**EDI.Collection** ファイルは、MapForce により使用できるようになっています。以下のようにテストします :

1. MapForce を開始します。
2. **「挿入」** メニューから **「EDI」** をクリックしてください。または、 **「EDI の挿入」** ツールバーボタンをクリックしてください。 下に表示されているような、新規コレクションの名前 "EDIFACT.Nanonull" を表示するダイアログボックスが開かれます。選択されると、コレクションは、次のメッセージの型のみを表示しています: "ORDERS"。



メモ: 開始する時点で、MapForce は、**...\MapforceEDI** ディレクトリの下の全ての兄弟サブフォルダーをスキャンし、"EDI.Collection" という名前のファイルを検索します。**EDI.Collection** ファイルを含む各フォルダー名がダイアログボックス内に表示されます。メッセージの型リストがコレクションファイルのコンテンツを表示します。この例では、ORDERS メッセージだけが含まれています。

グローバルなカスタマイズとインラインなカスタマイズ

この例の目的は、CTA (Contact Information) セグメントを再定義することです。CTA セグメントは、1つのフィールド (F3139) と1つのコンポジット (C056) により構成されています。例えば、"Mr" または、"Mrs" などの)タイトルデータを保管するには、C056 コンポジットに X1000 と呼ばれる新規のフィールドを追加します。カスタマイズを行うには複数の方法があります:

- **EDSD.segment** ファイルをカスタマイズすることにより、**グローバルに**カスタマイズを行うことができます。全てのメッセージ内の全てのセグメントは、コンポジット C056 を使用し、新しい要素を含むまたは参照します。
- **ORDERS.Config** ファイルをカスタマイズすることにより、**インラインに**カスタマイズを行うことができます。現在のメッセージ内のカスタマイズされたセグメント (CTA) のみが新規の要素を含みます。

グローバルなカスタム化

新規の X1000 フィールドにグローバルにアクセスを作成する場合、**EDSD.Segment** ファイルのみに変更が加えられます。コンポジット C056 を使用する全てのメッセージ内の全てのセグメントは、新しい要素を含むまたは参照します。

コンポジット C056 をグローバルに再定義します:

1. **EDSD.Segment** ファイルをテキストまたは XML エディター内で開き、**「構成 | 要素 | コンポジット | C056」**にナビゲートします。

```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
</Composite>
```

2. 次のラインを C056 セグメントの "F3412" の下に挿入します:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

`Composite` 定義は以下のようになります:

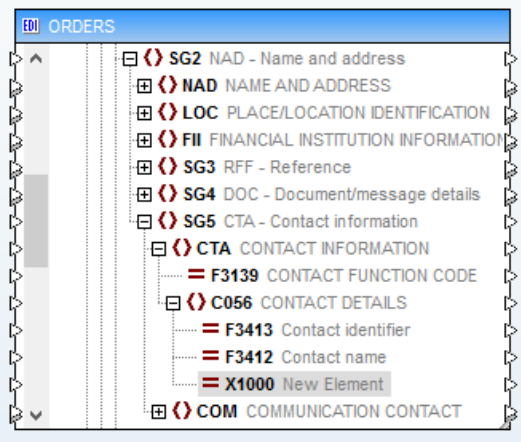
```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
```

```
  <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
</Composite>
```

上のコードリストでは、 "ref" 属性。を使用するセグメントの他のフィールドに対して、新規のX1000 フィールドが "name" 属性を使用して定義されています。他の2つのフィールドは、ファイルの最初に定義されており、参照されているだけです。

C056 を使用する全てのメッセージで新規のX1000 フィールドを使用することができます。MapForce 内の新しいフィールドを以下のようにプレビューすることができます：

1. **「挿入」**メニューから**EDI** をクリックしてください。または、 **「EDI の挿入」** ツールバーボタンをクリックしてください。
2. "EDIFACT.Nanonull" コレクションをクリックして、ORDERS メッセージを選択します。
3. ソースEDI ファイルを選択するように問われると、 **「スキップ」**をクリックします。ORDERS が、マッピングウィンドウ内で表示されています。
4. コンポーネントヘッダーをクリックし、 **「Ctrl + F」**を押して、"X1000" を検索テキストとして入力します。 **「次を検索」**をクリックし、**Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056** の下に存在する新規のX1000 要素の次の発生に移動します。



インラインのカスタマイズ

新規のX1000 フィールドへローカルに(*または、インラインで*)アクセスするには、**ORDERS.config** ファイルのみに変更が加えられなければなりません。この場合、**現在の**メッセージ内の再定義されたCTA セグメントは、新しい要素を含むまたは参照します。すなわち、CTA セグメントは、*カスタム*新規のフィールドX1000 を持つ再定義されたコンポジット C056 を含むようにローカルで再定義されます。

コンポジットC056 をローカルで再定義する：

1. **ORDERS.Config** ファイルをテキストまたはXML エディター内で開き、 `Group name="SG5"` にナビゲートします(または、**SG5** を検索します)。

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact
information">
   <Segment ref="CTA"/>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

2. ライン`<Segment ref="CTA"/>` を以下のラインと置き換えます：

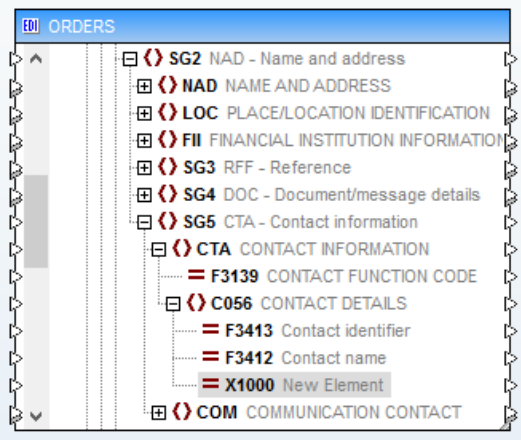
```
<Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
    <Data ref="F3139" minOccurs="0"/>
    <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE
DETAILS">
          <Data ref="F3413" minOccurs="0"/>
          <Data ref="F3412" minOccurs="0"/>
          <Data name="X1000" type="string" maxLength="35"
minOccurs="0" info="New Element"/>
    </Composite>
</Segment>
```

Group 定義は以下のようになります：

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact
information">
   <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>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

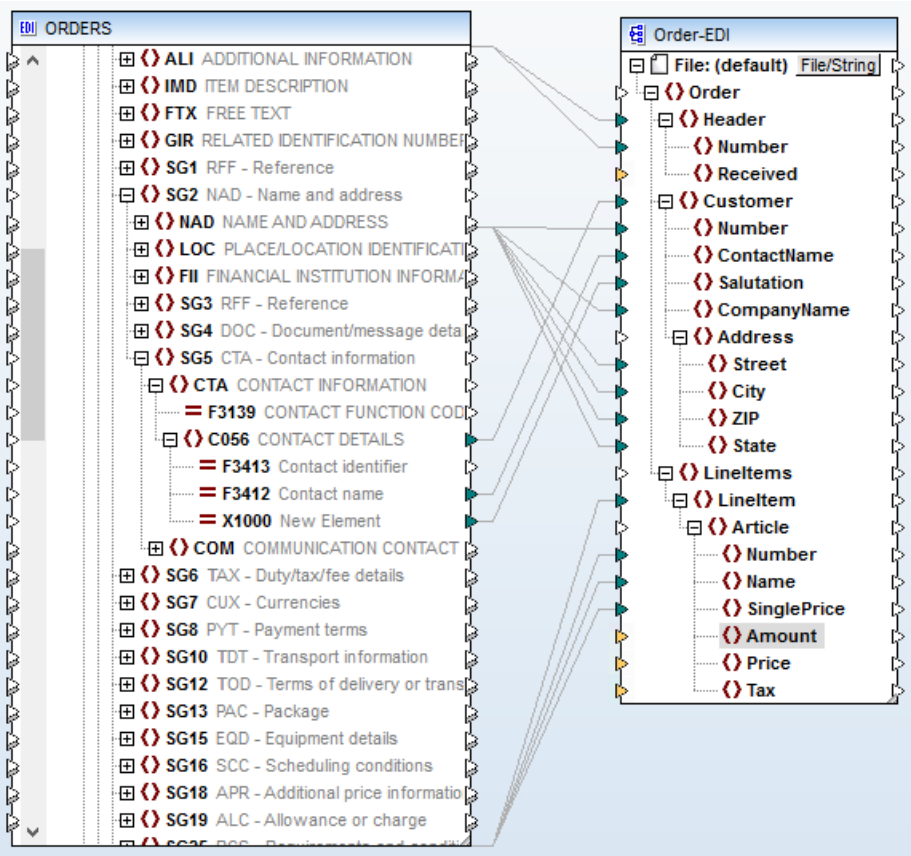
MapForce 内の新しいフィールドを以下のようにプレビューすることができます：

1. **「挿入」**メニューから**EDI** をクリックしてください。または、 **「EDI の挿入」** ツールバーボタンをクリックしてください。
2. "EDIFACT.Nanonull" コレクションをクリックして、ORDERS メッセージを選択します。
3. ソースEDI ファイルを選択するように問われると、**「スキップ」**をクリックします。ORDERS が、マッピングウィンドウ内で表示されています。
4. **Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056** にナビゲートすると、新規のX1000 要素を確認することができます。



マッピング内でカスタマイズされたメッセージを使用する

上記の手順に従った場合、カスタム EDIFACT 構成 をマッピング内で使用することができます。上記のカスタム EDIFACT 構成を使用するマッピングの例は、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダー内の**Orders-Custom-EDI.mfd** です。この中では、**ORDERS-Custom.EDI** ファイルが**Order-EDI** スキーマにマップされます。カスタム フィールドがEDI 構造に追加され、`X1000` ターゲットスキーマ内の`Salutation` アイテムにマップします。



上記の手順に従った場合、このマッピングを開き、マッピングの実行に成功する事ができます。しかしながら、カスタマイズの手順に従わなかった場合、このマッピングを以下のとおり実行することができます：

1. "**EDIFACT.Nanonull**" と呼ばれる新規のディレクトリを "**C:\Program Files\Altova\MapForce2017\MapForceEDI**" ディレクトリの下に作成します。32-ビット MapForce を 64-ビット オペレーティングシステム上で作動している場合、パスを**C:\Program Files (x86)\Altova\MapForce2017\MapForceEDI** に調整してください。
2. **EDIFACT.Nanonull.zip** ファイルを（**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** にある新規のディレクトリで解凍します。
3. **Orders-Custom-EDI.mfd** を開きます。

マッピングの結果をプレビューするには、**「出力」** タブをクリックします。

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
  <Header>
    <Number>ABC123456XYZ</Number>
  </Header>
  <Customer>
    <Number>92</Number>
    <ContactName>Ted Little</ContactName>
    <Salutation>Mr</Salutation>
    <CompanyName>24h Pizza</CompanyName>
    <Address>
      <Street>Long Way</Street>
      <City>San-Francisco</City>
      <ZIP>34424</ZIP>
      <State>CA</State>
    </Address>
  </Customer>
  <Customer>
    <Number>92</Number>
    <ContactName>Michelle Butler</ContactName>
    <Salutation>Mrs</Salutation>
    <CompanyName>Nanonull, Inc.</CompanyName>
```

MapForce 内の他のマッピングデザインファイルと類似して、EDI マッピングは、MapForce 外で実行されることもできます。例えば、他のサーバーにデプロイし、MapForce Server または、FlowForce Server で実行する、または、プログラムコードを生成し、マッピングを実行することなど。この例のスコープのために、C++ コードを生成してマッピングを実行することを選択した場合、"C056Type" クラスからアクセスすることのできる "X1000Type" という名前のクラスが生成されます。

7.5.12.12 例 :ANSI X12 トランザクションのカスタム化

この例は、非標準、または、変更された X12 フォーマットを処理するために、MapForce のカスタム化について説明しています。このサンプルには、サンプルマッピングが存在し、以下のパスで見つけることができます :**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Orders-Custom-X12.mfd**。このマッピングを次のカスタム化を行う前に開いた場合、MapForce は、正確に検証できないことに注意してください。この理由は、マッピングが、使用中のマシン上にまだ存在しない カスタム X12 構成を参照しているからです (この例の中でこの構成は作成されます)。

Orders-Custom.X12 ファイルからデータを読み取るサンプルのマッピングは、**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリ内にあります。このファイルをテキストエディターで開くと、カスタム化された N2 セグメントが使用されていることに気がつきます。具体的には、6列目では、「++**Mrs**」エントリを含みます :

```
ISA+00+          +00+          +ZZ+SenderID       +ZZ+ReceiverID
GS+PO+SenderID+ReceiverID+20060308+182347+1+UN+050112'
ST+850+12345'
BEG+00+SA+ABC123456XYZ++20040430'
N1+1 +Nanonull, Inc.++123'
N2+Michelle Butler++Mrs'
N3+119 Oakstreet Suite 4876'
N4+Vereno+CA+29213'
PO1++1++7.2'
PO3+1 +++7.2++1+US'
PID+1++++Pizza Pepperoni'
TXI+1 ++9'
AMT+1+720'
PO1++2++13.2'
PO3+1 +++6.6++2+US'
```

上記のような カスタム X12 ファイルからデータをマップする前に、カスタム X12 構成が作成される必要があります。このサンプルでは、この例では、個人のタイトルに対応するように N2 セグメントに新規フィールドを持つように拡張するために、デフォルトの X12 構成をカスタマイズします。参照のために、カスタマイズの最終結果は、ZIP アーカイブ (**X12.Nanonull.zip**) として **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** ディレクトリ内にあります。下の手順は、ZIP ファイルが **C:\Program Files\Altova\MapForce2017**

\MapForceEDI ディレクトリディレクトリ」に既にアンパック済みであると想定しています。

カスタマイズファイルのセットアップ

1. MapForce インストールディレクトリ (**C:\Program Files\Altova\MapForce2017**) を開きます。 32-ビット MapForce が 64 ビット オペレーティングシステム上で作動している場合は、パスを**C:\Program Files (x86)\Altova\MapForce2017\MapForceEDI** に調整してください。
2. **..\MapForceEDI** ディレクトリの下に**X12.Nanonull** という名前のディレクトリを作成します。
3. …**\MapForceEDI\X12** フォルダーから次のファイルを**X12.Nanonull**フォルダーにコピーします。

   - **EDI.Collection**
   - **Envelope.Config**
   - **850.Config**
   - **X12.Segment**
   - **X12.Codelist**

   各ファイルの目的に関する詳細に関しては、次を参照してください EDI 構成ファイル.

メモ: ファイルの属性を編集できるように **読み取り書き込み」**に変更してください。

EDI.Collection ファイルの構成

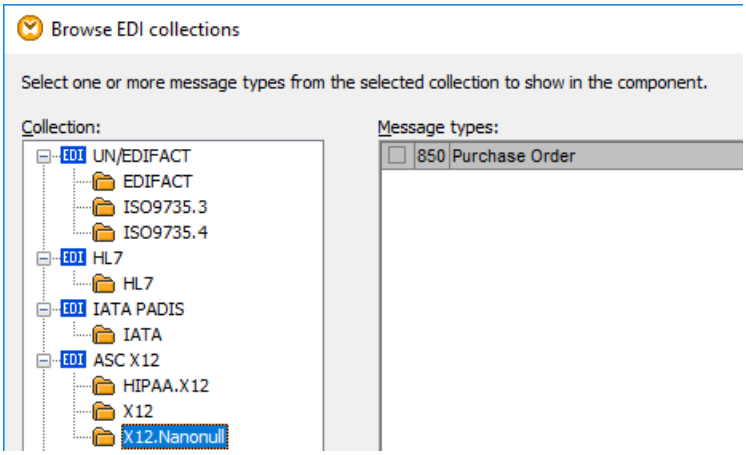
1. **EDI.Collection** ファイルを、XML エディターで開きます (例えば、XMLSpy)。
2. "Purchase Order" メッセージ以外のすべての "Message" 要素を削除します。 <Messages> タグが保持されていることを確認してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Release>6040</Release>
      <Agency>X12</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="850" File="850.Config" Description="Purchase Order"/>
</Messages>
```

3. ファイルを保存します。

新規の**EDI.Collection** ファイルは、MapForce により使用できるようになっています。以下のようにテストします :

1. MapForce を起動します。
2. **「挿入」**メニューから、**「EDI」**をクリックします。 または、 **「EDI の挿入」** ツールバーボタンをクリックします。 下に表示されているような、新規コレクションの名前 "X12.Nanonull" を表示するダイアログボックスが開かれます。選択されると、コレクションは、次のメッセージの型のみを表示しています: "Purchase Order"。



メモ: 開始する時点で、MapForce は、**...\MapforceEDI** ディレクトリの下の全ての兄弟サブフォルダーをスキャンし、"EDI.Collection" という名前のファイルを検索します。**EDI.Collection** ファイルを含む各フォルダー名がダイアログボックス内に表示されます。メッセージの型リストがコレクションファイルのコンテンツを表示します。この例では、ORDERS メッセージだけが含まれています。

グローバルとインラインのカスタム化

この例の目的は、N2 "Additional Name Information" セグメントを再定義し、個人のタイトルを保管するカスタム X1000 データ要素が含まれるようになります (例えば、"Mrs")。カスタム化の前に、N2 は、1つのデータ要素 "F93 Name" のみを構成しています。カスタマイズを行うには複数の方法があります:

- **X12.Segment** ファイルをカスタマイズすることにより、**グローバルに**カスタマイズを行うことができます。"グローバル" とは、すべてのセグメントでは、N2 を使用するすべてのトランザクションのすべてのセグメントで、新しい要素を含むまたは参照することを意味します。
- **850.Config** ファイルをカスタマイズすることにより、**インラインに**カスタマイズを行うことができます。"ローカル" とは、現在の トランザクション内のカスタム化されたセグメント (N2) に新規の要素が含まれることを意味します。

グローバルなカスタム化

新規のX1000 フィールドにグローバルにアクセスを作成する場合、**X12.Segment** ファイルのみに変更が加えられます。 すべてのセグメントでは、N2 を使用するすべてのトランザクションで、新規の要素を含む、または、参照します。

N2 をグローバルに再定義します:

1. **X12.Segment** ファイルを、XML エディターで開き、**構成 | 要素 | Segment name="N2"** に移動します。

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
</Segment>
```

2. F93 の下に次を挿入して、ファイルを保存します。

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

セグメント定義は以下のようになります:

```
<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>
```

メモ:

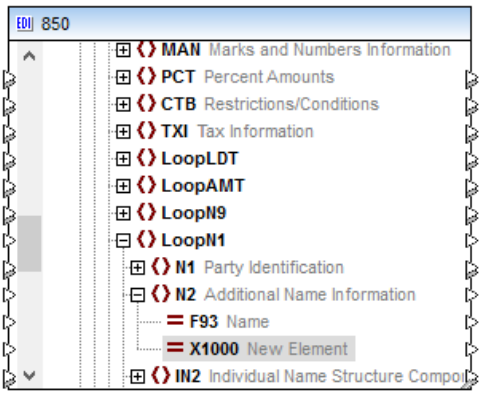
- 新規のX1000 フィールドは、ref 属性を使用セグメントの他のフィールドに対して、name 属性を使用して定義されています。F93 フィールドは、**X12.Segment** ファイルの始めで定義されており、1度のみ参照されます。
- F93 のmergedEntries 属性は、同じ名前を持つこの要素の2つの連続した発生を指定しています。これは、ASC X12 仕様内で定義されているようにこのフィールドの定義を示しています。このため、ここでカスタム化されているソースEDI ファイル内では、N1 セグメントは以下のようになります:

```
N2+Michelle Butler++Mrs'
```

上のラインでは、"Michelle Butler" と "Mrs" の間の 2つの連続した "+" セパレーターに注意してください。N2 セグメントは、カスタム "X1000" データ要素 ("Mrs") が後に続く "+" セパレーターが後に続く 2番目の (空の) "F93" 要素が後に続く "+" セパレーターが後に続く "F93" データ要素 ("Michelle Butler") により構成されているからです。 mergedEntries 属性に関する詳細は、次を参照してください 複数の連続したデータ要素の扱い

セグメント N2 を使用する全てのトランザクションで新規のX1000 フィールドを使用することができます。MapForce 内の新しいフィールドを以下のようにプレビューすることができます:

1. **「挿入」** メニューから **「EDI」** をクリックしてください。または、 **「EDI の挿入」** ツールバーボタンをクリックします。
2. "X12.Nanonull" コレクションをクリックし、 "Purchase Order" メッセージを選択します。
3. ソースEDI ファイルを選択するようにプロンプトされると、 **「スキップ」** をクリックします。 850 コンポーネントが、マッピングウィンドウ内で表示されています。
4. コンポーネントヘッダーをクリックし、 **「Ctrl + F」** を押して、"X1000" を検索テキストとして入力します。 **「次を検索」** をクリックし (**Envelope/Interchange/Group/Message_850/Loop_N1/N2** の下に存在する) 新規のX1000 要素の次の発生に移動します。



ローカルでのカスタム化

新規のX1000 フィールドへローカルにアクセスするには、 **850.Config** ファイルのみに変更が加えられなければなりません。この場合、"850 - Purchase Order" トランザクション内のセグメント N2 のみが、新規のX1000 フィールドを含む、または、参照します。

N2 をローカルで再定義する:

1. **850.Config** ファイルをテキストまたはXML エディター内で開き、`Group name="LoopN1"` (または **LoopN1** を検索します)。

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
   <Segment ref="N1"/>
   <Segment ref="N2" minOccurs="0" maxOccurs="2"/>

   ... ...

</Group>
```

2. ライン`<Segment ref="N2" minOccurs="0" maxOccurs="2"/>` を次のラインと置き換えます:

```
<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>
```

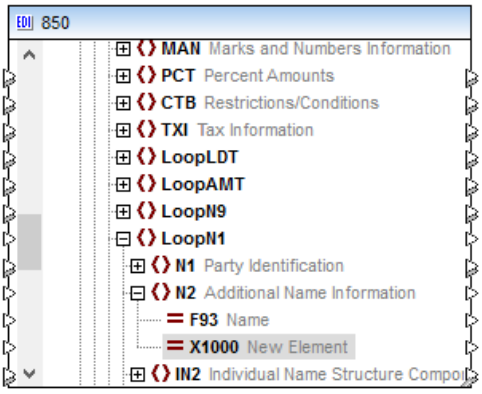
`Group` 定義は以下のようになります:

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
   <Segment ref="N1"/>
   <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>

   ... ...

</Group>
```

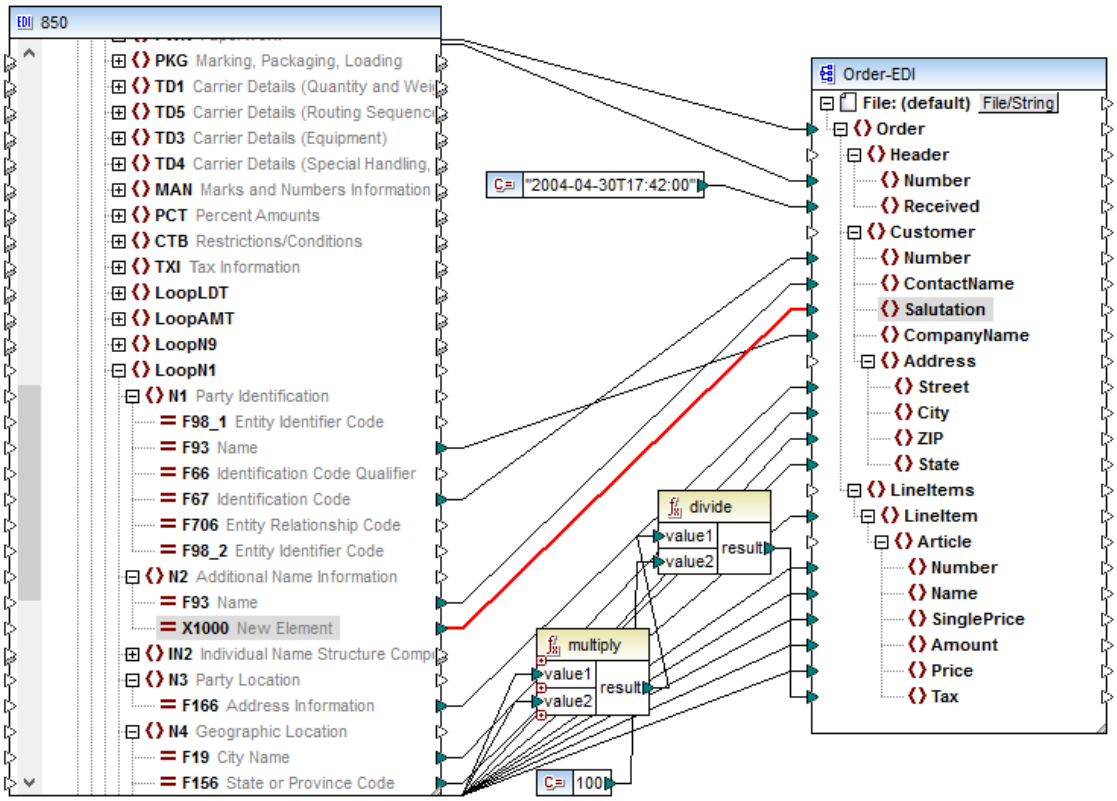
グローバルなカスタム化のために説明された命令を使用して、MapForce 内で新規のフィールドをプレビューすることができます。



マッピング内でカスタム化されたメッセージを使用する

上記の命令に従うと、マッピング内でカスタムX12 構成を使用できます。上記のカスタムEDIFACT 構成を使用するサンプルマッピング **Orders-Custom-X12.mfd** は **<Documents>\Altova\MapForce2017**

\MapForceExamples\Tutorial フォルダー。は内で見つけることができます。このサンプルは、**ORDERS-Custom.X12** ファイルを**Order-EDI.xsd** スキーマにマップします。ターゲット スキーマ内の`Salutation` アイテムに上記のステップで追加されたカスタム`X1000` フィールドがマップされます。



Orders-Custom-X12.mfd

上記の説明に従うと、このマッピングを開き、実行することができます。マッピングの結果をプレビューするには、**「出力」** タブをクリックします。期待されているとおり、Salutation ターゲットアイテムがX12 ファイルのカスタムX1000 データ要素より抽出された値 "Mrs" と共に表示されます。

```
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>
```

7.5.13 構成ファイルを使用して EDI 検証をカスタマイズする

このセクションは、EDI メッセージの検証をMapForce 内でカスタム化して、以下の目的を達成する方法について説明します:

- 検証ルールが標準により完全に実施することのできないフィールドの検証を弛緩し、外部の標準を利用して拡張する場合もあります。

  例えば、定義済みのコードの値に追加して、ISO 外部標準、または、国連欧州経済委員会 (UN/ECE) の勧告により指定されたコードの値を受け入れるEDI データ要素があるとします。この場合、このようなEDI フィールドを処理する場合、MapForce を構成して、外部コードが発生すると (検証エラーではなく)警告を返し、マッピング処理を再開することができます。詳細に関しては、次を参照してください:フィールドの検証の弛緩。

- カスタム構成ファイルをMapForce に適用し、上記にようなフィールドを検証します。

  例えば、UN/EDIFACT D.16A 標準のデータ要素 4405 を検証するには、国連欧州経済委員会 (UN/ECE) 勧告第 24 号 (「貿易および運送のステータスコード」により定義された検証ルールを含む、カスタム構成ファイルを追加する必要があるかもしれません。国連欧州経済委員会 (UN/ECE) 勧告第 24 号内で定義されたコード値は、EDIFACT 標準 に対して外部となります (このため、MapForce により与えられるEDIFACT 構成ファイルのデフォルトのセットの一部ではありません)。詳細に関しては、次を参照してください:外部ソースからコードの値を適用する。

- フィールドが使用されている箇所、または、特定のメッセージのために、ローカルで使用されている全てのメッセージのためにグローバルに検証を行います。

  グローバル、または、ローカルに検証を構成することは可能です。グローバルな検証は、全てのメッセージ内のフィールドでの全ての発生に影響を与えます。ローカル (または「インライン」)検証は、特定のメッセージのみに適用することができます。詳細に関しては、次を参照してください:メッセージレベル (インライン)でコードの値を定義する。

- フィールドレベルだけではなく、文字レベルでも検証が必要なフィールドのために高度な検証ルールを定義します。

  UN/EDIFACT D.16A のVERMAS (大容量のメッセージの検証)などの特定のメッセージは、(データ要素 8155 などの)高度な検証ルールを必要とするフィールドを含んでいます。このフィールドは、ISO 6346 内で指定されているコード値を受け入れる可能性があります。ISO 6346 などのフィールドを検証するために、構成ファイルを使用して、フィールド内で特定の文字が特定の場所で発生するかをチェックするように、MapForce をカスタム化することができます。詳細に関しては、次を参照してください:文字の構成 レベルの検証。 (ISO 6346 は、ここで例として記述されており、MapForce 構成ファイルは既にISO 6346 のための検証ルールを既に含んでいます)。

7.5.13.1 フィールドの検証の弛緩

標準がフィールド内のコードの値を完全に適用しない場合、外部の標準または勧告により定義されたコードを許可する場合があります。EDIFACT D16A を例としてみましょう。フィールドの一部は、外部 ISO 標準内、または、UN/ECE 勧告で定義されたコードの値を許可します。例えば、データ要素 4405は、146 コードの値を定義しますが、346 コードの値を指定する外部 UN/ECE 勧告第 24 号の使用も許可しています。

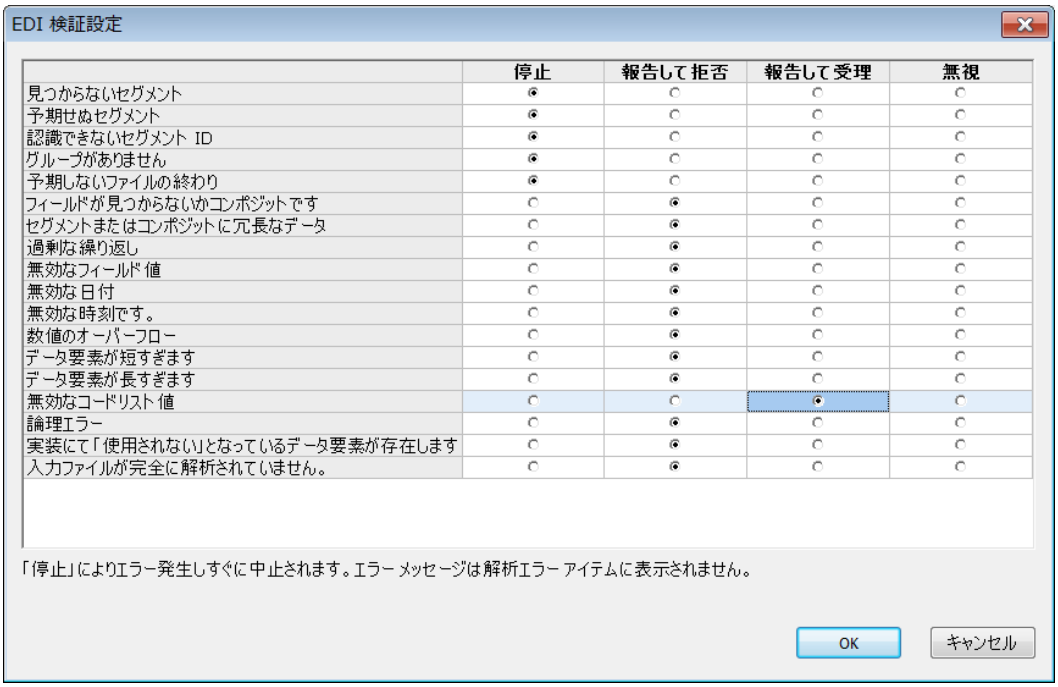
MapForce 内では、フィールドのためのコードの値は (以降、「codelist」ファイルとして称される)codelist 拡張子を持つ構成ファイルに対して検証されます。MapForce がEDI ファイルを処理する際、フィールドがその種類のEDI ための .codelist ファイル内で指定されているコードの値を持たない場合、通常の場合検証エラーが表示されます。 (これは、無効なフィールドの値を無視するために、デフォルトの検証オプションが変更されていないと想定されています。次を参照してください:EDI コンポーネント検証)。**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ内で各 EDI 種類のための codelist ファイルを使用することができます。例えば、**C:\Program Files\Altova\MapForce2017\MapForceEDI\EDIFACT** 内で使用することのできる、**UNCL.codelist** ファイルは、UN/

EDIFACT フィールド内のフィールドのための使用できるコードの値です。

デフォルトの codelist ファイルは、対応するEDI 標準に従い、フィールドのために使用できるすべてのコードの値を含みます。コードの値が標準により明確に定義されておらず、外部の標準により強制されている場合、検証は不完全とみなされます。具体的には、EDI 標準が (UN/EDIFACT D16A のデータ要素 4405 などの)特定のフィールドのためにコードの値のリストを強制しない場合、マッピングの実行中のフィールドの処理は、通常検証エラーを表示します。具体的には、MapForce (または、MapForce Server、やC#、C++、またはJava プログラム、などのEDI マッピングを作動する実行エンジン)は、次のフォーマットでエラーを報告します:**{Value} is not a legal value for field {Field}**。このエラーは、 codelist ファイルで定義されている既存のコードリストの値のリスト内でフィールドの値が見つからなかったことを意味します。

マッピングの処理が上記のエラーなどにより中断されることを避けるためには、次を行います:

1. EDI 検証設定から検証の振る舞いを変更します (例えば、EDI 検証設定 ダイアログボックス上の 無効な コードリスト値」の横の **報告して拒否」**チェックボックスを選択して (EDI 検証設定を構成するには、EDI コンポーネント設定 ダイアログボックスの **検証」**ボタンをクリックしてください。)



2. 構成ファイから固定の検証を弛緩する方法は以下に表示されています。

MapForce は、フィールドに対して 未完成」検証ルールが存在するとマークするメカニズムを適用します。これにより、MapForce (または、MapForce Server、やC#、C++、またはJava プログラム、などのEDI マッピングを作動する実行エンジン)は、フィールドが処理される際、エラーメッセージを作成しません。その代わりに、警告が表示され、マッピングの処理が再開されます。すなわち、 未完成な」検証は、を意味します。外部で適宜されているコードの値ではなく、MapForce デフォルトの codelist ファイル内で定義されているコードの値のみが、検証のために考慮されます。

複雑性を追加することなく、柔軟性を追加するには、MapForce により与えられているデフォルトの codelist 構成ファイル特定のフィールドで不完全な検証が存在すると扱うことです。

デフォルトでは、次のフィールド値が外部仕様内で定義されているコードの値を含むと、MapForce は、(エラーではなく)警告を作成します:

- UN/EDIFACT D.16A のデータ要素 4053

- UN/EDIFACT D.16A のデータ要素 4277
- UN/EDIFACT D.16A のデータ要素 4405
- UN/EDIFACT D.16A のデータ要素 8179

警告は、このフィールドの検証が不完全であることを通知するために作成されます。不完全な検証は、正確では内データがマッピングにより生成されるために発生する場合があります。これを回避するには、カスタムの codelist ファイルを MapForce に対して使用し、フィールド全体の検証をおこなうことが奨励されます。カスタムの codelist ファイルは、を外部のコードの値を MapForce が理解できるフォーマットで定義し、フィールド全体の検証を行います。詳細に関しては、次を参照してください:外部ソースからコードの値を適用する。

メモ: データ要素 8155は、検証のために、外部 標準 (ISO 6346 標準)に依存しています、このフィールドのために、MapForce は、個別の codelist ファイルを使用して、全体の検証行うこともできます。次を参照してください:外部ソースからコードの値を適用する)。

必要であれば、特定のフィールドに対して、不完全な検証ルールが適応されるようにMapForce をカスタム化することができます。MapForce に、フィールドに不完全な検証ルーツが存在すると命令するには、以下を行います:

1. **C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ (それぞれのEDI 種類のために存在するサブディレクトリ)内の.codelist ファイルを検索し、XMLSpy などのXML エディターで開きます。す。
2. 必要とされるデータ要素に対応する<Field> 要素を検索し、(下でハイライトされている)属性 Incomplete="1" を追加します。例:

```
<Field Name="F8179" Length="8" Type="Alphanumeric" Incomplete="1">
   <Title>Transport means description code</Title>
   <Description>Code specifying the means of transport.</Description>
   <Note>1 Use UN/ECE recommendation 28.
</Note>
   <Values>
      <Value Code="1">
         <Title>Box tricycle</Title>
         <Description>Tricycle with a basket for delivering goods.</
Description>
      </Value>
   </Values>
</Field>
```

メモ: このコンテンツを変更するには、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリに書き込むパーミッションが必要です。

「未完成」とマークされるフィールドに対して処理の必要条件が警告ではなくエラーを表示することを必要とする場合、属性 Incomplete を「1」ではなく、値「0」に変更してください。この方法で、フィールド検証が厳密に行われ、フィールドの値が、codelist ファイル内で定義されている値のセットの一部ではない場合、警告ではなく)エラーが表示されます。

7.5.13.2 外部ソースからコードの値を適用する

MapForce により、デフォルトで提供されたEDI 構成ファイル内では、データ要素の一部は、フィールドの検証の弛緩で説明されているように不完全な検証が存在するように構成されています。これは、これらのデータ要素の可能なコード値は、EDI 標準自身の外部である仕様に依存するからです。

不完全な（または、弛緩されている）フィールドの検証が受け入れられない、または、希望されない場合、MapForce を構成してフィールド全体を検証することができます。このアプローチは、MapForce 内で使用できるデフォルトのものに加え、カスタムの codelist カスタムに対しても使用することができます。 codelist ファイルの役割は、デフォルトで EDI 標準内で定義されていないために MapForce 構成ファイルに存在しないコードに値を与えます。これは、外部の標準、または、勧告内で定義されているコードの値に通常適用されます。

カスタムの codelist ファイルは、検証される使用できるコードの値のためにエントリを必要とします（場合によっては数百個のエントリになる場合があります）。UN/EDIFACT D16A のデータ要素 4405 は1つの例です。UN/EDIFACT 標準で定義されているコードの値に追加して、このフィールドは、UN/ECE 勧告 24（「貿易および運送のステータスコード」内で指定されているコードの値も許可します。346個のコードを持つ次を参照してください：http://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf）。勧告内の 346個のそれぞれのコードのためのエントリを含む外部のコードの値すべてに対応するため、カスタムの codelist ファイルを作成する必要があります。ですから、特定のビジネスのケースで不完全な検証の使用が許可されている場合、不完全な検証は簡単な解決方法です（次を参照してください：フィールド検証の弛緩）。不完全な検証は、マッピングの出力内で無効なデータが生成されることを許可する可能性があるため、代わりにフィールド全体の検証が奨励されます。

フィールド全体を検証するには、外部の標準内で定義されているコード値で定義されているコードの値を使用します。次の手順を追ってください：

1. MapForce により与えられるデフォルトの codelist ファイルのコピーを作成します。codelist ファイルは、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ内で、各 EDI 種類のために使用することができます。例えば、カスタムの codelist ファイルを UN/EDIFACT メッセージを検証するために追加すると、**C:\Program Files\Altova\MapForce2017\MapForceEDI\EDIFACT** 内の **UNCL.Codelist** ファイルをテンプレートとして使用します。

メモ： コンテンツを変更するには、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリへ書き込む権限が必要です。

2. ファイルは、他の MapForce 構成ファイルから参照されるため、新規に作成された codelist ファイルの名前を短くかつ分かりやすいファイル名に変更します。このサンプルのスコープのために、カスタムの codelist ファイルには、**UNECE24.codelist** という名前が与えられます。 .
3. XMLSpy などの XML エディターを使用して、`<Codelist>` と `</Codelist>` タグ間に含まれる全てのコンテンツを新規のファイルから削除します。ファイルは以下のようになります：

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>

...

   </Codelist>
</Config>
```

4. `<Codelist>` の下の新規の `<Field>` 構成は下に示されています。このサンプルでは、UN/EDIFACT. のデータ要素 4405 ために外部のコードの値を追加します。これ以外の場合、フィールドの属性を特定のケースで必要とされるとおり置き換えます。MapForce により提供されるデフォルトの codelist ファイル内の各フィールドの名前、長さ、型を確認することができることに注意してください。（このサンプルでは、**UNCL.Codelist** です）.

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
```

```
         <Values>

...

         <Values>
      </Field>
   </Codelist>
</Config>
```

5. <Values> の下に、各外部のコードのために新規の<Value> を以下のように追加します。このサンプルでは、UN/ECE 勧告 24 貿易および運送のステータスコード」内で定義されている最初の 2 つのコードを追加します（次を参照してください：http://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf）。

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
         <Values>
            <Value Code="1">
               <Title>Arrival, completed</Title>
               <Description>The goods/consignment/equipment/means of
transport has arrived.</Description>
            </Value>
            <Value Code="2">
               <Title>Loading, authorized</Title>
               <Description>Authorization to load has been given.</
Description>
            </Value>
         <Values>
      </Field>
   </Codelist>
</Config>
```

この段階では、カスタムの codelist ファイルの準備はできていますが、MapForce にバウンドされてはいません。このサンプルのスコープのために、2 つの外部コード（「1」と「2」）のみが追加されていますが、実社会のシナリオでは、カスタムの .codelist ファイルは、外部の標準により指定される全てのコードの値を含むよう期待されます。

最後に、カスタムの codelist ファイルが MapForce により認識される前に、以下を行う必要があります：

1. データ要素 4405 には不完全な検証がもう存在しないため、MapForce はフルの検証を適用するように命令されなければなりません。具体的には、Incomplete="1" 属性は、**UNCL.codelist** ファイル内で、このフィールドのために変更される必要があります（属性が削除されるか、または、Incomplete="0"に設定されなければなりません）。このため、**UNCL.codelist** ファイルを下記の通り変更します。

```
...

<Field Name="F4405" Length="3" Type="Alphanumeric" Incomplete="0">
         <Title>Status description code</Title>
         <Description>Code specifying a status.</Description>
         <Note>1 For transport status, use UN/ECE Recommendation 24.</
Note>
```

```
...
```

2. **UNCL.codelist** ファイルを変更して、新規に作成された**UNECE24.codelist** ファイルにレファレンスを追加する方法は、は以下のとおりです：

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="UNECE24.codelist"/>
   <Codelist>

....
```

カスタムファイル内で定義されたコードの値からのフィールド 4405 を検証するための、MapForce 構成ファイルの適用が完了しました。このアプローチは、外部の標準をベースにしたフィールド全体の検証を達成するために、他の EDI 種類のために使用することもできます。

外部コードリストを MapForce により提供される既存の codelist ファイルを編集して定義することもできますが、(このサンプルでは **UNCL.codelist** です)、これは奨励されません。外部コードリスト in カスタムファイル内の外部コードリストを定義して、デフォルトの MapForce 構成ファイルへの変更を最低限にとどめておくことが奨励されます。これは、将来 EDI フォーマットの新しいバージョンを移行する際に簡単に作業を行うためです。

上記の命令は、フィールド 4405 を使用する可能性のあるセグメントのために、フィールド検証をグローバルにします。一部のケースでは、特定のメッセージのみにカスタムのコードリストを適用する場合があります (すなわち、構成は、全てのメッセージのためにグローバルに適用されてはならず、特定のメッセージのためのみにインライン、または、ローカルに適用されます)。に関する詳細は、「インライン」アプローチの使用に関しては、次を参照してください：メッセージレベル (インライン)でコードの値を定義する。

7.5.13.3 メッセージレベル (インライン)でコードの値を定義する

前のトピックの外部ソースからコードの値を適用するでは、(特定のフィールドを使用しているメッセージに適用する)カスタムのコードリストをグローバルに使用してフィールド検証を構成する方法について説明がされました。特定のメッセージのみに検証を適用する必要がある場合は、(特定のメッセージのために)ローカルに適用し)カスタムのコードリストをインラインに追加することが可能です。以下の手順に従ってください：

1. XML エディター内で必要なメッセージに対応する **{Message}.config** ファイルを開きます。**{Message}.config** ファイルは、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリ内の必要とされる EDI 種類の名前に対応するサブディレクトリ内で検索することができます。
2. 必要とされる の下から<Data> 要素を検索し、カスタムの コードの値を<Values> の子として追加します 例：

```
<Group name="Message_270-B1" info="Health Care Eligibility Benefit
Inquiry" maxOccurs="unbounded">
      <Segment name="ST" info="Transaction Set Header">
        <Condition path="F1705" value="005010X279A1" />
        <Data ref="F143" info="Transaction Set Identifier Code">
          <Values>
            <Value Code="270" />

[>>>> custom values can be added here...]

          </Values>
```

```
        </Data>
        <Data ref="F329" info="Transaction Set Control Number" />
        <Data ref="F1705" info="Implementation Convention Reference" />
      </Segment>
```

以下の点に留意してください:

- コンテンツを変更するには、**C:\Program Files\Altova\MapForce2017\MapForceEDI** ディレクトリに書き込む権限が必要です。
- フィールドが(上のコードリスティング内で説明されているような)ローカルで定義された、および、(外部ソースからコードの値を適用するで表示されている codelist ファイルのような)グローバルで定義されたコードの値 の両方を有する場合、ローカルのコードの値がグローバルなコードの値としても定義されていることを確認してください。これ以外の場合、検証エラー*** が挙げられます:**{Value} is not a legal value for field {Field}**。
- インラインの (local) 検証は、フィールドレベル(すなわち、そのフィールドでの可能な値がリストとして表示されることができるレベル)で検証が必要な場合使用することができます。インラインの検証は、文字レベルで複雑な検証を必要とするフィールドのためには使用することができません。(UN/EDIFACT 標準のデータ要素 8155 に適用することのできるISO 6346 内で定義されているような)フィールドが複雑な検証ルールを文字レベルで必要とする場合、外部ソースからコードの値を適用するで示されるとおり、コードの値をグローバルに追加し、検証を文字レベルの検証の構成で示されるとおり構成します。

*** 実際の検証の結果はEDI 検証設定 ダイアログボックス内のルール 無効なコード リストの値」のために構成された設定を考慮することに注意してください (このダイアログボックスを開くには、EDI コンポーネント設定 ダイアログボックスの **検証** ボタンをクリックしてください)。

7.5.13.4 文字の構成 -レベルの検証

多くの場合、フィールドの検証は、マッピングランタイムでのフィールドの値をMapForce 構成ファイルにより定義された値のリストと比較することを意味します。例えば、フィールドの値が構成ファイル内で定義されている使用できる値の 1 つに一致する場合、有効とみなされます。

しかしながら、特定のフィールドは、更に高度な検証ルールを必要とする場合があります。これは、検証のロジックが使用できる値の簡単なリストとして表示することができないことを意味します。複雑な検証を必要とするフィールドのサンプルは、UN/EDIFACT D16A のVERMAS (Verification of Mass) メッセージ内で使用されている、データ要素です。このフィールドは、UN/EDIFACT 標準内で定義されているコードの値だけではなく、ISO 6346 標準により指定されている外部コードも受け入れます。 ISO 6346 標準は、文字レベルで検証ルールを定義しています。具体的には以下が挙げられます:

- 最初の文字は次の 1 つであることができます:1、 2、 3、 4、 A、B、C、D、E、F、G、H、K、L、M、N、P
- 2 番目の文字は次の 1 つであることができます:0、 2、 4、 5、 6、 8、 9、 C、D、E、F、L、M、N、P
- 3 番目、 4番目の文字は次の組み合わせの 1 つであることができます:G0、G1、G2、G3、など。 (全ての値を表示することはできません)。

MapForceIt より高度な検証 ロジックを適用することもできます。この場合、MapForce は、データ要素 8155 を、UN/EDIFACT コードの値だけではなく、ISO 6346 標準に適用することができるルールにも従い、検証します。

MapForce 構成パースペクティブからは、上記の 3 つのルールは、 3 つの個別の値のリストとして示されています。 最初のリストおよび 2 番目のリストは、 1 文字で、 3 番目のリストは、 2 文字です。 3 つ全てのリストは、有限で値は既知です。この情報は、MapForce 構成を使用してデータ要素の値を検証することを可能にします。

文字レベル検証の構成を確認するには、**ISO6346.Codelist** ファイルを**C:\Program Files\Altova\MapForce2017\MapForceEDI\EDIFACT** ディレクトリから開きます。 このファイルはデフォルトで与えられており、MapForce は、上記のISO 6346 ルールに従い、データ要素の検証を行うように構成されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1"
Offset="1">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="1"/>

            ..............

            <Value Code="N"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1"
Offset="2">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="0"/>

            ..............

            <Value Code="F"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="2"
Offset="3">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="G0"/>

            ..............

            <Value Code="A0"/>
         </Values>
      </Field>
   </Codelist>
</Config>
```

ISO6346.Codelist 構成 file

メモ: 上記のサンプルのコードリスティングの値の一部は、簡素化のために削除されました。

上のコードリスティング内で、 <Field> 要素は、 3回定義されています。これにより、フィールドのコンテンツは文字レベルで検証することが可能になります。 具体的には、<Field> が発生する都度、上記のルールのうち 1つを検証します。これを達成するには、各 <Field> には (上のコードリスティングでハイライトされているように) 複数の属性が存在します :

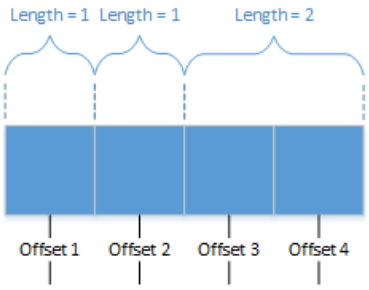
- Name - この属性は、検証される値のリストは同じフィールドに属することを示します。すなわち、これは 3つのリストをこの場合、 「F8155」 同じフィールドにバインドします。
- ListName - この属性は、値のリストに名前を与えます。同じ名前を持つ値のリストは、 1つのコードリストとしてグループ化されます。上のサンプルでは、 3つ全てのリストは、同じListName="ISO6346" を持ちます。これは、文字レベルの検証に必要です。グループ内の各リストのLength とOffset 属性により指定されている実際の文字の検証に関しては、下で説明されています。

- `Length` - この属性は、検証される文字の長さを指定します（文字の開始のポジションは、`Offset` 属性により定義されているフィールド内に存在します）。文字の長さは、それぞれの `<Field>` により異なります：最初と2番目の値のリストは、1文字で、3番目のリストは2文字です。
- `Offset` - この属性は、MapForce（または、外部の実行エンジン）が検証されるデータフィールドから文字を読み込む開始の位置を指定します。最初の文字は、オフセット1を持ち、2番目の文字はオフセット2を持つなどです。実際の文字数は、`Length` 属性により定義されます。

> `Length` 属性が定義されている場合、`Offset` 属性も定義される必要があります。また、`Length` と `Offset` の値は0より大きくなければなりません。

上のコードリスティングでは、以下の長さ-オフセット組み合わせが定義されています：

- `Length="1", Offset="1"` - この組み合わせは、オフセット1から開始するフィールドの1文字を検証します。
- `Length="1", Offset="2"` - この組み合わせは、オフセット2から開始するフィールドの1文字を検証します。
- `Length="2", Offset="3"` - この組み合わせは、オフセット3から開始するフィールドの2文字を検証します。



`<Values>` の下で定義されているコードの値と組み合わされている各長さ-オフセット組み合わせは、検証の条件を文字レベルで示しています。フィールド検証が成功するためには、マッピングの作動時、すべての文字レベル条件が満たされている必要があります（これは全てが同じ `ListName` 属性値持つためです）。すなわち、データ要素 8155 を含むメッセージを検証する場合、MapForce は次のロジックを適用します：

- フィールド文字は、長さ-オフセット組み合わせの一致のために、構成ファイル（この場合、**ISO6346.Codelist**）内で定義されているコードの値のリストに対して検証されます。例えば、フィールドの最初の文字は、以下のリスト内の1つであることができます：1, 2, 3, 4, B, C, G, H, L, M, N.
- 検証を成功させるには、フィールドの値は、文字列での全ての条件を満たす必要があります（この場合、3つの条件が満たされていなければなりません）。
- 条件の一つが満たされない場合、フィールドの値は有効ではなく、MapForce（または、MapForce Server、や C#、C++、または Java プログラムなどの）EDI マッピングを実行する実行エンジン内で検証エラーが生成されます。
- フィールドは、使用できるコードの値の1つ以上のリストを持つことができます。有効とみなされるためには、フィールドの値は、リストの1つに存在する必要があります。

7.5.14 JSON

JSON (JavaScript Object Notation) をインスタンスとスキーマファイルにマッピング上のソースまたはターゲットコンポーネントとして追加することができます。MapForce は JSON Draft 04 スキーマをベースにして (http://tools.ietf.org/html/draft-zyp-json-schema-04) JSON ファイルを読み込み、または書き込みます。

JSON ファイルをマッピングに追加する場合、MapForce は、スキーマまたはインスタンスファイルかを決定するためにファイルが `$schema` プロパティを持つかチェックします。プロパティが存在する場合、"http://json-schema.org/schema#"

または "http://json-schema.org/draft-04/schema#" であり、ファイルはスキーマとしてロードされます。それ以外の場合、MapForce はファイルがスキーマかインスタンスかを通知します。ロードされたファイルが JSON インスタンスファイルであることを確認すると、MapForce は、スキーマを参照するか、または、自動的に生成するかを問います。MapForce は、JSON スキーマをコンポーネントの構造を構築するために使用します。

MapForce は、JSON スキーマファイルを **schema.json** 拡張子（例 :**Example.schema.json**）で識別します。このドキュメントが作成されている時点で、JSON スキーマファイルに名前をつけるための公式な規則は存在しません。

MapForce 内の JSON コンポーネントは、通常 JSON インスタンスファイルの構造を持ちます。すなわち、JSON コンポーネント内のノードの構造は、JSON インスタンスファイルの構造を反映します。基本的な JSON 型は、MapForce 内で JSON コンポーネントとして以下のテーブルのように表示されます。

型	MapForce 表示
配列	[]
ブール値	01
Null	∅
数値または整数	#
オブジェクト	{}
文字列	"AB"

特別なケースは、以下のとおりです。

ケース	説明
additionalProperties、*patternProperties*	ノード は、`additionalProperties` プロパティが `true` に設定されている、または、スキーマ内に存在しない、オブジェクトの下の JSON コンポーネントに表示されます。これにより、明示的にスキーマにリストされていないプロパティにマップ、または、プロパティからマップすることができます（例 :JSON からCSV へマッピングも参照）。 このノードは、`patternProperties` プロパティのために表示されることもできます。
サブタイプ	JSON スキーマは、オブジェクトと配列（`anyOf`、`allOf`、`oneOf`）のために サブタイプを定義することを許可します。MapForce は、JSON インスタンスファイル内で直接の表記を持たない特別な構造のノード（ ）を使用してこのようなサブタイプを表示します。 マッピングが JSON ファイルから読み込んでいる場合、このようなサブタイプノードは、現在の入力の値が、サブ型スキーマに従い有効な場合のみ値を与えます。 マッピングが JSON ファイルに書き込む場合、正しいサブタイプが選択されていることを確認してください。複数のサブタイプへの入力は、オブジェクトプロパティの重複に繋がる可能性があり、このため、出力 JSON ファイルが無効になる可能性があります。
同じ場所での複数の型	JSON スキーマは、同じ場所で複数の型が派生します。このような場合、MapForce コンポーネントは、この場所で派生するすべての基本型のための個別の構造ノードを表示します。
型の名前	MapForce は、存在する場合、JSON スキーマ内の `title` と `description` プロパティの型を "型 "と "注釈 "フィールドに表示します。`title` が存在しない場合、MapForce は、`$ref` プロパティの URI の一部を型の名前として使用する場合があります。

ケース	説明
複合型 アイテムの型を含む配列	JSON スキーマ内でアイテムの異なる種類を持つ配列の場合 (例 :文字列と番号)、MapForce は、それぞれのアイテムの型のための "アイテム" ノードを表示します。JSON ファイルに書き込む場合、これにより、異なる種類のアイテムを含む配列を作成することができます。
タプルとして定義される配列	JSON スキーマ内のポジションにより割り当てられる型を持つアイテムが配列に存在する場合、MapForce は、ゼロをベースにしたアイテムのインデックス個別の構造ノード (例えば、**item[0]**、**item[1]**、など)としてを表示します。JSON ファイルに書き込む場合、これにより、配列内のそれぞれの個別のアイテムの型をゼロをベースにしたインデックスで指定することができます。

JSON サポートについてのメモ
マッピング内でJSON コンポーネントを使用する場合、ターゲット言語を "ビルドイン" に設定することが必要です (変換言語の選択を参照)。JSON コンポーネントはコード生成 言語 (C#、C++、Java) 内で使用することはできず、 また、JSON コンポーネントは現在以下をサポートしません :

- ユーザー定義関数内の複雑型パラメーター
- 中間変数 コンポーネント .

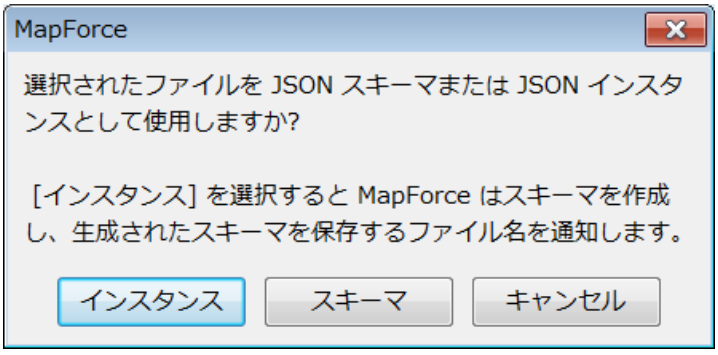
以下を参照してください :

- JSON ファイルをマッピングコンポーネントとして追加する
- JSON コンポーネント設定
- 例 :JSON からCSV へマッピング

7.5.14.1 JSON ファイルをマッピングコンポーネントとして追加する

JSON ファイルをマッピングコンポーネントとして追加する

1. **「挿入」** メニューから **「JSON スキーマ/ファイル」** をクリックします。ダイアログボックスがJSON ファイルの型を選択するようにプロンプトします (スキーマまたはインスタンス)。



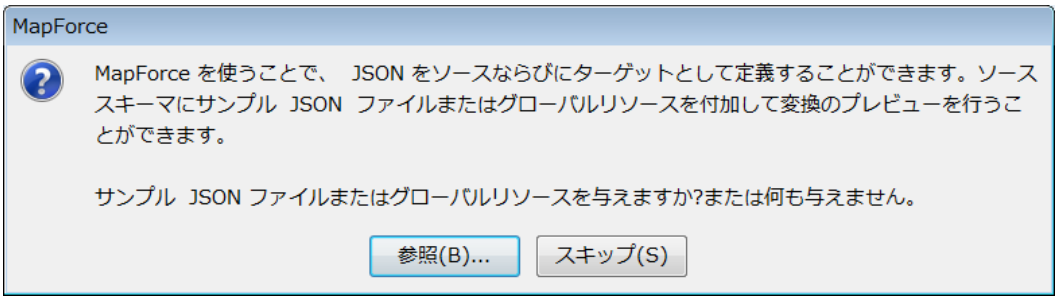
2. 必要に応じて、 **「スキーマ」** または **「インスタンス」** を選択します。

> JSON インスタンスファイルのサイズはシステムで使用できるメモリを超えてはなりません。

「インスタンス」 を選択した場合、MapForce インスタンスファイルをベースにしたスキーマを自動的に生成し、スキ

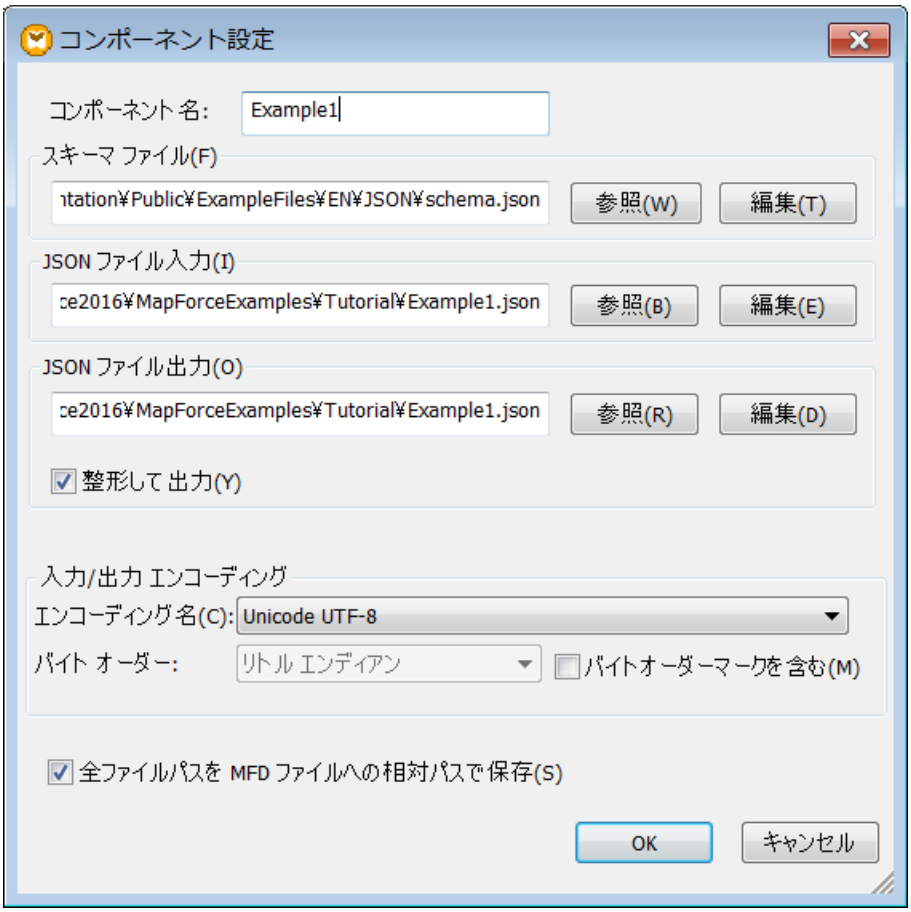
ーマを保存する場所をプロンプトします。

スキーマ を選択した場合、MapForce はサンプルJSON ファイル または グローバルリソースを指定するようにプロンプトします。 (グローバルリソースを参照)。変換をプレビューすることは必須です。



7.5.14.2 JSON コンポーネント設定

JSON コンポーネントの設定を変更するには、JSON コンポーネントヘッダーを右クリックして、 **プロパティ** をクリックします。(または、 JSON コンポーネントヘッダーをダブルクリックします)。



JSON コンポーネント設定 ダイアログボックス

使用することのできる設定は、以下のとおりです .

コンポーネント名	JSON コンポーネントの名前の表示を変更することができます。
スキーマファイル	スキーマファイルのファイル名とパスを指定します。 ファイルのロケーションを変更するには、 **参照...** をクリックして、新しいファイルを選択します。 JSON エディター (例 :XMLSpy) 内でファイルを編集するには、 **編集...** をクリックします。
入力 JSON ファイル	入力 JSON インスタンスのファイル名とパスを指定します。 ファイルのロケーションを変更するには、 **参照...** をクリックして、新しいファイルを選択します。 JSON エディター (例 :XMLSpy) 内でファイルを編集するには、 **編集...** をクリックします。
出力 JSON ファイル	MapForce Server によりマッピングが実行されると、JSON ターゲットインスタンスファイルが配置されるファイル名とパスを指定します。 **出力** タブから出力を保存する際のデフォルトのロケーションです。 ファイルのロケーションを変更するには、 **参照...** をクリックして、新しいファイルを選択します。 JSON エディター (例 :XMLSpy) 内でファイルを編集するには、 **編集...** をクリックします。
整形出力	構造化された表示のため、マッピングが実行される際、JSON ドキュメントを再フォーマットします。各子ノードは、単一タブ文字づつそれぞれの親のオフセット。
エンコード	出力インスタンスファイルの次の設定を指定することを許可します : • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。 JSON ファイルは、UTF エンコードを持つことが期待されます。 (http://tools.ietf.org/html/rfc7159#section-8.1 を参照)。他のエンコードは、非標準と考えられます。
JSON5 の使用	コンポーネントが入力 JSON ファイルを読み取るデータが JSON5 構文を使用する場合このチェックボックスを選択します。 ターゲットコンポーネント内での、チェックボックスの選択は、引用符で囲まれていない、オブジェクトキーが有効な ECMAScript5 識別子である場合、 json5 ファイルを作成するように命令します。次を参照します : JSON5 へのサポート。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design

	(.mfd) ファイルの場所に相対したファイルパスを保存します。この設定は、以下のファイルに影響を与えます： • JSON スキーマファイル • JSON 入力ファイル • JSON 出力ファイル コンポーネント上で相対パスを使用するも参照してください。

7.5.14.3 JSON5 へのサポート

JSON5 は、JSON ファイルを手動で作成するユーザーが使用しやすいように提案された JSON への拡張子です。重要な点は、JSON5 は JSON の公式な後継ではありません（詳細に関しては、、次を参照してください：http://json5.org/）.

JSON5 ファイルは、 json5 拡張子を使用します。しかしながら、このドキュメントの作成時では、 json5 ファイルのための公式の MIME 型破損罪しません。 json5 拡張子は、しかしながら、マッピングへそのようなファイルが追加されると、MapForce により認識されます。 マッピングを FlowForce Server にデプロイすると、 json5 ファイルの MIME 型は `application/json` としてレポートされます（これは、json5 ファイルのために MIME 型が存在するからです）（次を参照してください：FlowForce Server にマッピングをデプロイする）。

必要条件

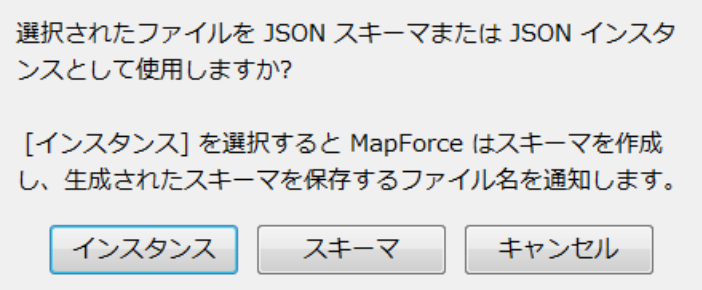
- JSON5 ファイルからのデータをマップするには、 （JSON5 構文を使用しない）有効な JSON スキーマを MapForce に提供する必要があります。下に示されるように、ファイルのために JSON スキーマが存在しない場合、MapForce はスキーマを自動的に生成します。MapForce により自動的に生成されるスキーマは（JSON5 ではなく）JSON スキーマです。XMLSpy を使用して、 json5 のために JSON スキーマを生成することは可能です。

制限

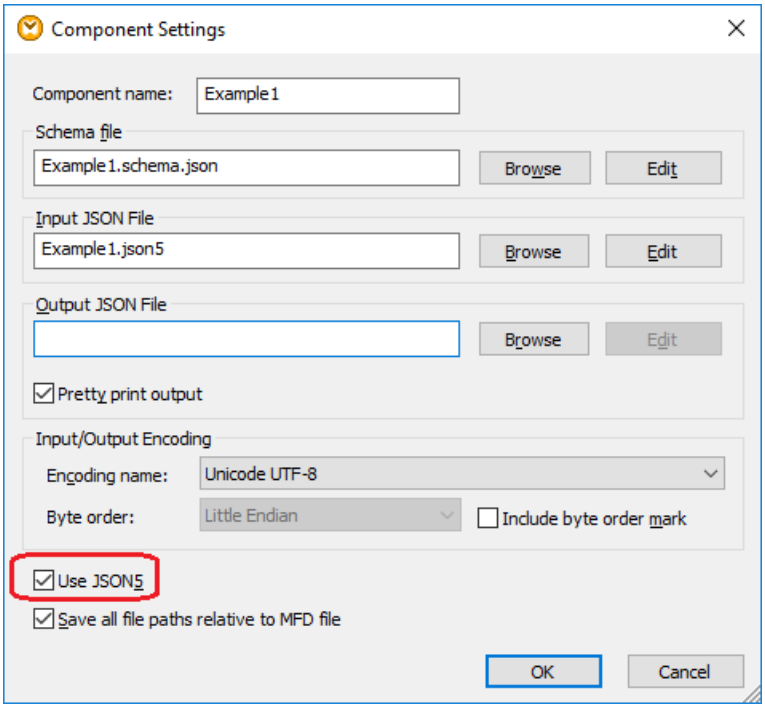
- JSON5 構文を使用する json5 ファイルの生成は、サポートされていません。生成された json5 ファイルは、しかしながら、JSON に準拠します。
- .json5 ファイルからのマッピングコメント、または、 json5 ファイルへのマッピングコメントはサポートされていません。
- MapForce は、厳格なデータ型と作業するため、 `Infinity`、`-Infinity`、`NaN`、または `-NaN` などの値を修飾、または、マップすることはできません。

.json5 ファイルからデータをマップする方法

1. **挿入** メニューから、**JSON** をクリックし、json5 インスタンスファイルを参照します（または、 ツールバーボタンをクリックします）。JSON スキーマファイルが存在し、 json5 インスタンスファイルを後で提供する場合は、有効な（JSON5 ではなく）JSON スキーマを参照してください。これはプレーンな JSON ファイルが作動すると同様に作動します（次を参照してください：JSON ファイルをマッピングコンポーネントとして追加する）。



2. JSON コンポーネントがマッピングに追加された後、ヘッダーを右クリックし、**プロパティ**を選択し、**「JSON5 の使用」**チェックボックスをコンポーネント プロパティから選択します (次も参照してください:JSON コンポーネント設定)。これにより、明示的にMapForce にインスタンスデータをそのように扱うように命令することができます。



MapForce を使用して、JSON5 ファイルを生成するには、**JSON5 の使用** チェックボックスを上に示されるように選択します。JSON コンポーネントを必要とされる入力データにマッピング上で接続します。 **JSON5 の使用** チェックボックスを使用する際には、MapForce は、生成されたJSON5 インスタンスが無効になる場合を除き、オブジェクトキーを引用符で囲みません。望ましいファイル拡張子 (例えば「json5」) は、上に示されるダイアログボックスで**出力 JSON ファイル** フィールド内に適切な値を入力することにより、管理することができます。

7.5.14.4 例: JSON から CSV へマッピング

このサンプルは、JSON インスタンスファイルからのデータをカンマで区切られたテキストファイルに変換するマッピングについて説明します。また、スキーマ内では定義されない、JSON インスタンスファイル内に存在する追加プロパティからの値をマップする方法について説明します。

このサンプルで使用されるJSON スキーマは以下に表示されるとおりです。`$schema` キーワードは、スキーマがDraft 04 JSON スキーマに関連するJSON インスタンスを検証することを示します。また、複数の`person` オブジェクトから構成される`people` 配列についても説明します。`people` オブジェクトが妥当であるためには、少なくとも1つの`person` オ

ブジェクトを含む必要があります。各 person オブジェクトには、name、age、および email がプロパティとして存在します。name と email は、型 string ですが、は型 integer です。また、name と email プロパティは必須で、age は任意です。

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "title": "people",
  "type": "array",
  "items": {
      "title": "person",
      "type": "object",
    "required": [
      "name",
      "email"
    ],
    "properties": {
      "name": {
        "type": "string"
      },
      "email": {
        "type": "string",
        "format": "email"
      },
      "age": {
        "type": "integer"
      }
    }
  },
  "minItems": 1
}
```

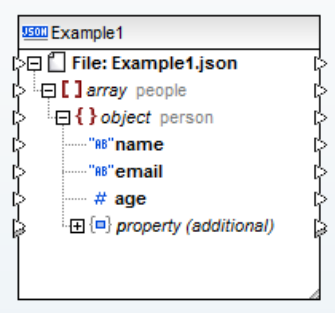
JSON インスタンスファイルは、CSV に変換されなければならない people レコードを含んでいます：

```
[
      {
            "name": "Alethia Alonso",
            "email": "altethia@example.com",
            "age": 35,
            "birthday": "4 July"
      }, {
            "name": "Klaus Mauer",
            "email": "klaus@example.com",
            "age": 57,
            "birthday": "31 August"
      }, {
            "name": "Natsuo Shinohara",
            "email": "natsuo@example.com",
            "age": 29
      }
]
```

ハイライトされたテキストは、最初と2番目の person がスキーマで定義されない birthday という追加プロパティを有することを示しています。しかしながら、スキーマが person オブジェクトのための additionalProperties プロパティを含まないため、JSON インスタンスは有効です。JSON スキーマ内に存在しない場合は、additionalProperties プロパティは、デフォルトの値である **true** を有します。これは、JSON インスタンス内のオブジェクトは、必要な数だけ追加プロパティを有し、有効であることができることを示します。

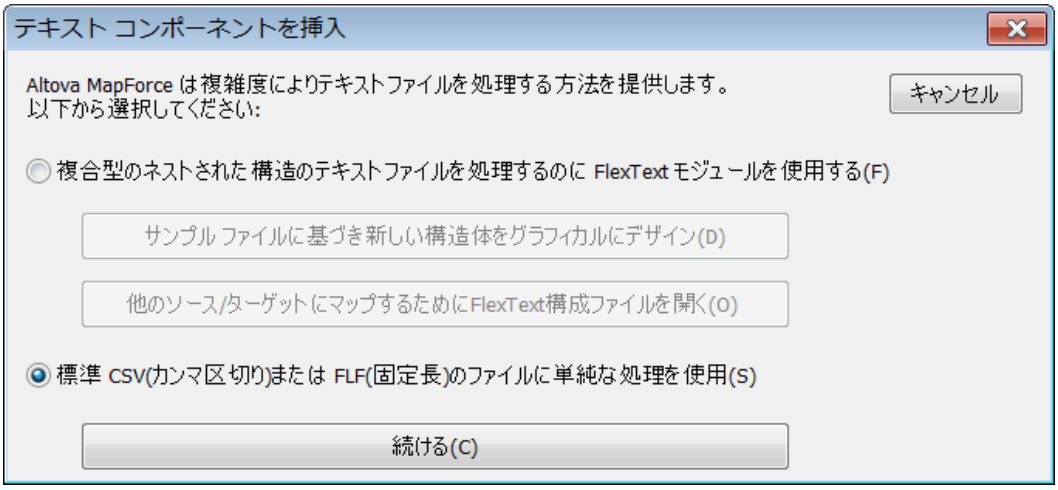
CSV マッピングへJSON を作成する

1. **挿入**」メニューから『**JSON スキーマ/ファイル**』をクリックします。 **<Documents>\Altova \MapForce2017\MapForceExamples\Tutorial** フォルダー内にある、**Example1.schema.json** ファイルを参照します。"マイ ドキュメント"フォルダー。インスタンスを指定するようにプロンプトされると、**Example1.json** ファイルを参照します。この時点では、MapForce コンポーネント外観は、以下の通りです：

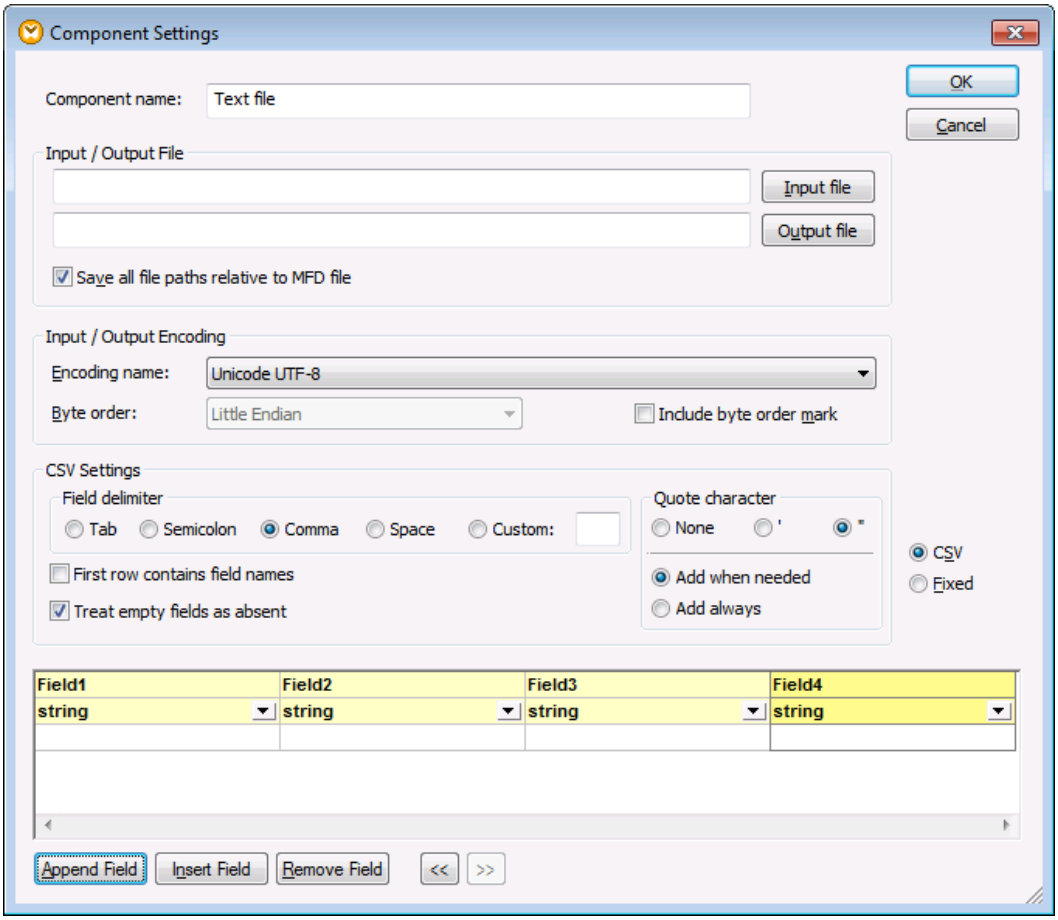


MapForce コンポーネントの構造は、**プロパティ (追加)** ノード以外、JSON ファイル自身に類似しています。このノードは、`person` オブジェクトの`additionalProperties` プロパティが不足している、または スキーマ内で**true** に設定されていることを示します。これは、スキーマがカスタム追加プロパティを含むことができることを意味し、MapForce が、オブジェクトの追加プロパティからマップする場合のために ノードを表示することを示しています (次のステップでこの手順について説明されています)。

2. **挿入**」メニューから『**テキストファイル**』をクリックします。

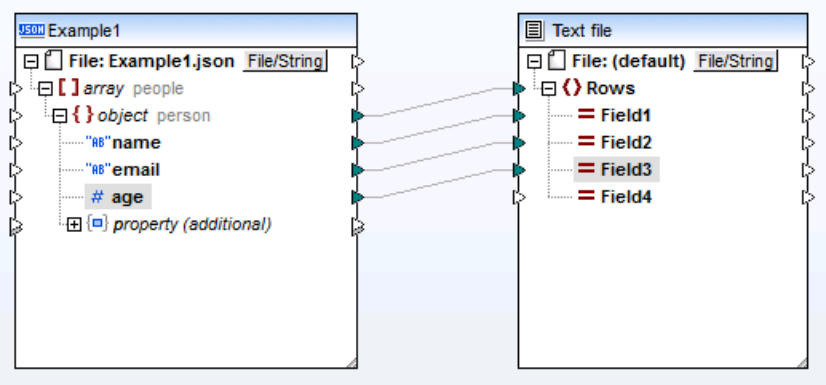


3. **標準CSV のための単純な処理**」を使用を選択して、 **続ける**』をクリックします。



4. **「フィールドの追加」**を4回クリックして、テキストコンポーネントにCSV フィールドを追加し **「OK」**をクリックします。これらのフィールドは、`person` オブジェクトの`name`、`email`、と`age` プロパティをマップします。4番目のフィールドは`birthday` 追加プロパティをマップします。

5. マッピングペインでは、JSON コンポーネントからの`person` オブジェクトをCSV コンポーネントの`Rows` ノードに接続します。また、person オブジェクトの`name`、`email` と`age` プロパティを テキストコンポーネントの最初の 3 つのフィールドに接続します。マッピングは、以下のようになります：



オプションで、CSV フィールドにオブジェクト プロパティ (すなわち、*name*, *email*, and *age*) と同じ名前を与えると、 **マッチする子要素の接続** オプションを使用して、すべての接続を自動的に行うことができます

(マッチする子要素の接続 を参照)。

この時点で、 **出力** タブをクリックすることにより、変換の出力をプレビューすることができます。結果は、以下のとおりです :

```
Alethia Alonso,altethia@example.com,35,
Klaus Mauer,klaus@example.com,57,
Natsuo Shinohara,natsuo@example.com,29,
```

出力が示すように、JSON ファイル内の各person オブジェクトでは、新規の行が作成され、オブジェクトプロパティがカンマは区切られます。これは期待されている振る舞いです。

しかしながら、スキーマ内には存在しないが、JSON インスタンスファイル内に存在するbirthday 追加プロパティはまだマップされていません。

JSON オブジェクトの追加プロパティからマップする

1. person オブジェクトの下の **プロパティ (additional)** ノードを展開します。**name** アイテムに注意してください。このアイテムにより、追加プロパティに名前を使用してアクセスすることができます。サンプル内では、追加プロパティの名前は、 "birthday" 、型は "文字列 "です。

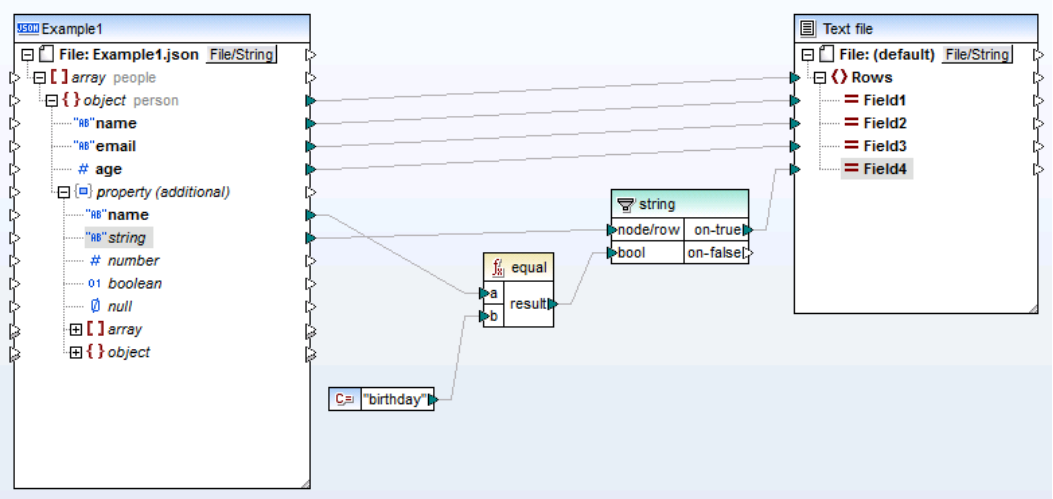
2. マッピングエリアに以下を挿入します :

   a. 値を持つ定数 "birthday"
   b. "Filter: node/row" コンポーネント
   c. logical 関数 **equal**

3. 以下の通りコンポーネントを接続することができます :

   - **equal** 関数の "a" 入力に対する追加プロパティの名前から。
   - **equal** 関数の "b" 入力に対する**birthday** 定数から.
   - "Filter: node/row" コンポーネントの**bool** 入力に対する**equal** 関数の出力から。
   - "Filter: node/row" コンポーネントの**node/row** 入力に対する**string** アイテムから。
   - テキストコンポーネントの **フィールド4** に対する"Filter: node/row" コンポーネントの**on-true** 出力から。

   コンポーネントをこのように接続すると、追加プロパティに**birthday** という名前を持つ追加プロパティをMapForce に検索するように命令します。文字列の値が見つかると、テキストコンポーネントの **フィールド4** にコピーします。



この時点で変換の出力をプレビューすると結果は、以下のとおりになります：

```
Alethia Alonso,altethia@example.com,35,4 July
Klaus Mauer,klaus@example.com,57,31 August
Natsuo Shinohara,natsuo@example.com,29,
```

出力が示すように、4番目のCSV フィールドは、型 "string" の追加プロパティのみの値 この場合、`birthday`、を含むようになります。また、3番目の個人は誕生日が記入されておらず、対応する場所のCSV 内では値を取得することはできません。

JSON インスタンス内に存在する追加プロパティをマップするためにこのテクニックを使用することができます。一般的なルールとして、マップする予定のすべてのプロパティがスキーマに含まれていることが奨励されます。しかしながら、JSON インスタンスがデータに依存する(固定されていない)プロパティを含む場合、このテクニックを使用します。

7.5.15 MS OOXML Excel 2007 以降のファイル

Altova Web サイト Excel へのからのマッピング

MapForce では、Microsoft Excel 2007 以降の (Open Office XML) ファイル (*xlsx) をソースならびにターゲットコンポーネントとしてマッピングすることができます。このフォーマットは、デフォルトの xlsx 拡張子に関連したExcel ワークブックのために最初にMicrosoft Office 2007で紹介されました。MapForce インターフェイス内とこのドキュメントでは、Microsoft Excel 2007 と以降のファイルは、通常 "Excel 2007+" ファイルと呼ばれます

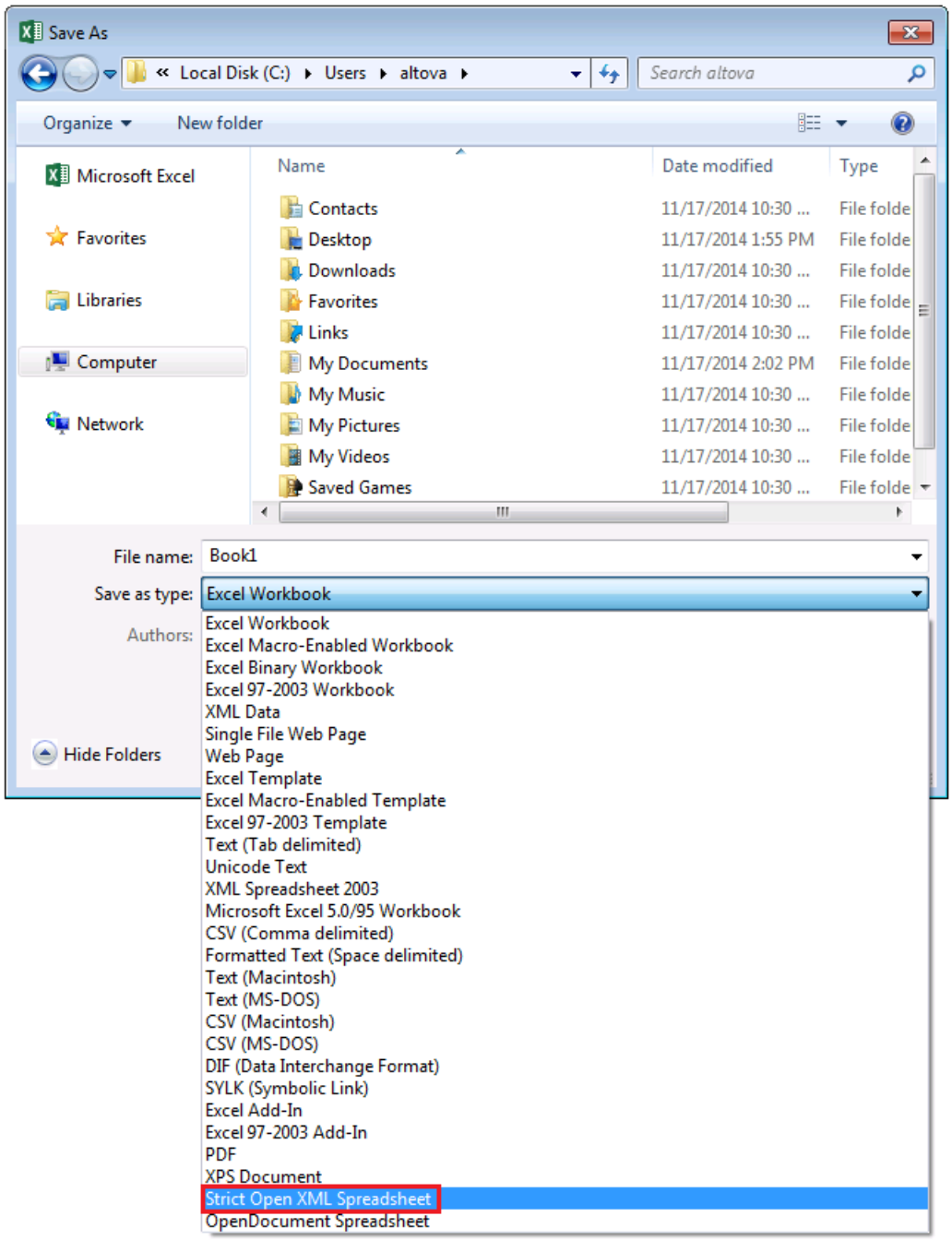
Microsoft Office 2013 は*Strict Open XML スプレッドシート*フォーマット (SO/IEC 29500 Strict) へのサポートの搭載を開始しました。*Strict Open XML スプレッドシート*フォーマットも xlsx 拡張子を有しますが、技術的には、更に厳格な検証ルールに準拠する特別のフォーマットです。

以下のテーブルは、MapForce 変換言語 によりサポートされるExcel 2007+ ワークブックから、または、へのデータの読み込みおよび書き込みを示したものです。 *Office Open XML* と*Strict Open XML スプレッドシート*フォーマットの違いに注意してください。

Microsoft Excel フォーマット	MapForce 言語	読み込み サポート	書き込み サポート
Office Open XML	BUILT-IN	はい	はい
	C#	はい	はい

	Java	はい	はい
	XSLT2	はい	いいえ
Strict Open XML スプレッドシート	BUILT-IN	はい	いいえ
	C#	はい	いいえ
	Java	はい	いいえ
	XSLT2	はい	いいえ

MapForce により生成された*Office Open XML ファイル*を*Strict Open XML スプレッドシート*フォーマットに変換する場合は、*ワークブック*をExcel 2013 で開き、*Strict Open XML スプレッドシート*として保存します。



Strict Open XML スプレッドシート フォーマットに保存する

MapForce 内のマッピングコンポーネントとして、Excel 2007+ ファイルの一般的な振る舞いは以下のとおりです：

- XBRL タクソノミファイルを含む、Excel 2007+ からのデータを MapForce にサポートされるコンポーネントにマップ または その逆、を行うことができます（Excel からXBRL へマップの例を参照）。
- Microsoft Excel 2007 または以降 が使用中のマシンにインストールされている場合、変換の出力をすぐにマッピングウィンドウの「出力」タブで確認することができ、ファイルに保存することができます。Excel 2007 または以降 が使用できない場合でも、Excel 2007+ ファイルからマップを行う または Excel 2007+ ファイルにマップを行う

ことはできますが、「出力」タブで、結果をプレビューすることはできません。ファイルは **出力**」メニューの **出力ファイルの保存**」をクリックすることにより、保存することができます。

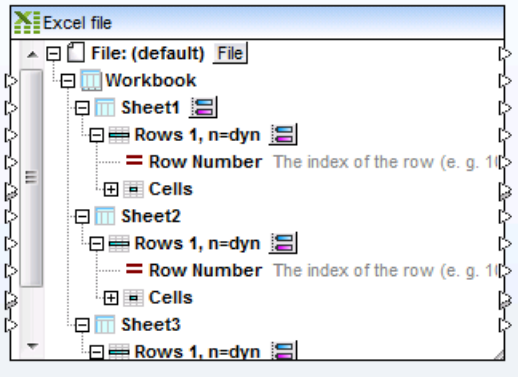
- 手動で出力を生成し保存する代替として、マッピングデザインをMapForce Server 実行ファイルにコンパイルすることができ、または、FlowForce Server にデプロイして、FlowForce Server ジョブにより必要とされる場合は、実行することもできます。更に詳しい情報に関しては、MapForce マッピングのコンパイルとMapForce マッピングのデプロイをそれぞれ参照してください。
- マッピングコード（上のサポートされる言語を参照して）を生成し、Visual Studio またはカスタムのアプリケーションからからマッピングを実行することができます。C# のためにコードを生成するには、OOXML サポートがVisual Studio .NET 3.0 または以降を必要とすることに注意してください。

このセクションは、以下のトピックを含みます：

- Excel 2007+ ファイルをマッピングコンポーネントとして追加する
- Excel 2007+ コンポーネントについて
- ワークシートの追加と削除
- 行の範囲の追加と削除
- セルの範囲の選択
- Excel 2007+ コンポーネント設定

7.5.15.1 Excel 2007+ コンポーネントについて

サンプルファイルを指定せずに、Excel 2007+ ファイルをマッピングエリアに追加すると、（Microsoft Excel ファイルをマッピングコンポーネントとして追加するを参照）、MapForce は、（下に表示される）3つのワークシートを含むデフォルトのコンポーネントを作成します。サンプルファイルを与えると、MapForce は、サンプルファイルを読み込み、必要とされるワークシートのみを作成します。



デフォルトのExcel コンポーネント

MapForce 内のExcel 2007+ コンポーネントの構造は、MapForce では、それぞれのセルをマップできるツリー構造で表示されている点が異なりますが、it Excel ワークブック内のデータ構造を反映します。w

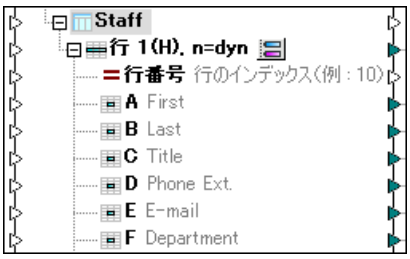
Excel 2007+ コンポーネントを他のコンポーネント型に接続する前に、MapForce に正確にどの行と列がマッピングで使用されるかを命令する必要があります。XML またはJSON などの他のMapForce コンポーネントとは異なり、Excel 2007+ ファイルには、MapForce がデータの構造を示すための明確なスキーマが存在しません。代わりに、定義することのできる設定がMapForce により与えられます：

- （ワークシート、範囲とテーブル、列、行など）どのデータが（ワークブックから読み込んでいる場合）ワークブックから明確に選択されるかを定義。
- MapForce が書き込む先のワークシート、列と行（ワークブックに書き込む場合）を定義。

ワークブック内の複数の箇所から読み込む為め、同じマッピングオペレーション内の複数の場所に書き込むために、にコンポーネントを構成することもできます。

必要とするコンポーネント構成設定を直接コンポーネントに対して使用することができます。 以下のテーブルを使用して、使用を開始してください。

コンポーネントアイテム	アイコン	説明
ワークブック		Excel ワークブックを表します。
ワークシート		Excel ワークシートを表します。最初のワークシートの横に表示される ボタンによりワークシートに関連した設定を指定することができます。
行		Excel 行の範囲を表します。ワークシート内に複数の行の範囲を追加することができます。これにより、同じマッピングオペレーション内のセルの複数の範囲から読み込む、または、範囲に書き込むことができます。 定義されたそれぞれの範囲のために、個別のデータ選択オプションを指定することができます。例：ソースExcel ファイル内のデータの量に従い、 1つの範囲は行 1から開始し、その行のすべての列を含む一方、他の範囲は、行 3から開始し、動的な数の行から構成されることができます。 以下を参照してください：一目で全ての範囲の設定を確認することができます。コンポーネントは、以下に説明されるように視覚的なヒントを提供します。 **行 *n*** *n* 行から開始する範囲を示します。 **行 *n*** *n* 行から開始する1行表示の範囲を示します。 **行 前+*n*** 前の範囲の後の、 *n* 行から開始する範囲を示します。 **行 *n*(h)** *n* 行から開始する範囲を示します。最初の行はヘッダー行として開始されます。 **n=*n*** *n* 行から構成される範囲を示します。 **n=dyn** 動的な範囲を示します。動的な範囲は無制限の数量の行を持つことができます。 各行の範囲の横に表示される ボタンにより、その範囲のための高度なデータの選択を指定することができます。
Cells		特定の行の全てのセル（列）を示します。このアイテムは、コンポーネントが全てのれるのために単一のセルに構成されている場合表示されます（MapForce のデフォルトの振る舞いです）。 または、以下に表示されるように、コンポーネントが各列を個別に表示するようにも構成できます。

コンポーネントアイテム	アイコン	説明
		
選択の変更		各ワークシート、または、行の横に表示される ボタンにより、そのコンテキストで意味をなす設定を指定することができます。このボタンを使用して、Excel コンポーネントの構造 を必要に応じて修正することができます。 例 Excel ファイル空データを読み取る場合、MapForce がデータを読み取るワークシート、行、列を指定することができます。Excel ファイルに書き込む場合、MapForce がデータを書き込むワークシート、行、列を指定することができます。

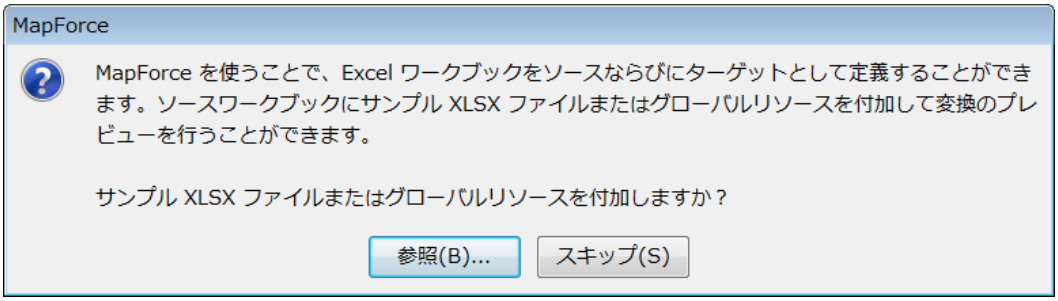
以下のセクションは Excel 2007+ コンポーネントに関連したオペレーションについて説明しています：

- ワークシートの追加と削除
- 行の範囲の追加と削除
- セルの範囲の選択
- Excel 2007+ コンポーネント設定

7.5.15.2 Excel 2007+ ファイルをマッピングコンポーネントとして追加する

Microsoft Excel 2007+ (*.xlsx) コンポーネントをマッピングエリアに追加する：

1. 以下を行います：
   - **「挿入」**メニューから **「Excel 2007+ ファイル」**をクリックします。
   - **「Excel 2007+ ファイルの入力」**（ ）ツールバーボタンをクリックします。



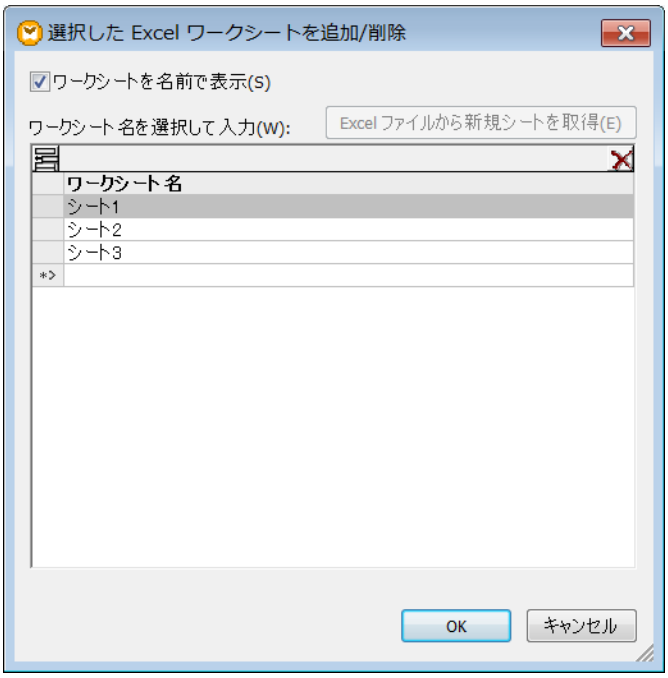
2. 以下を行います：
   - Microsoft Excel ワークブックからデータをマップする場合、 **「参照」**をクリックしデータをマッピングする xlsx ファイルから選択します。MapForce は、サンプル Excel ファイルを使用して、ワークシート名とセルの範囲を読み込みます。Excel ファイルをグローバルリソースとして定義した場合、グローバルリソースダイアログボックスからこれらを選択することもできます（グローバルリソースを参照）。サンプルファイルを後で追加する場合、 **「**

キップ] をクリックします。

- Microsoft Excel ワークブックへデータをマップする場合、 **[スキップ]** をクリックします。デフォルトでは、マッピング変換が実行されると、MapForce は、**xlsx-mapforce.xlsx** という名前の出力 Excel ファイルマッピングフォルダー内に生成します。必要な場合は、出力ファイルの名前をExcel 2007+ コンポーネント設定から変更することができます (Excel 2007+ コンポーネントについてを参照)。

7.5.15.3 ワークシートの追加と削除

Excel 2007+ コンポーネントのワークシートを手動または入力 xlsx ファイルをロードすることにより、追加または削除することができます。ワークシートノードの横の ボタンをクリックします。



Add/Remove Excel ワークシートselections ダイアログボックス

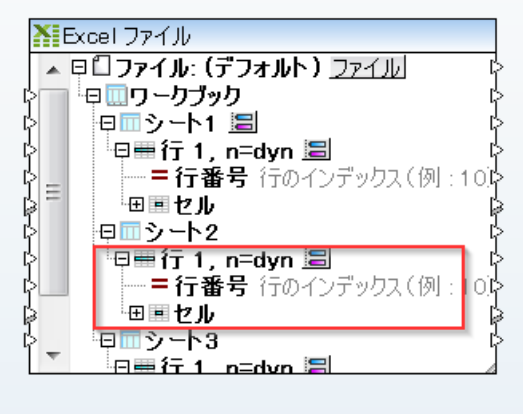
ワークシートに適用することのできるオプションは、以下のとおりです。

ワークシートを名前別に表示する	ワークブック内のそれぞれのワークシートが異なるレイアウトを有する場合、また、このためにMapForce 内で個別に表示される場合、このオプションが選択される必要があり。 ワークブック内のワークシートに同様の構造が存在する場合、このオプションを非アクティブに設定することができます。こうすることにより、MapForce は、ワークシートアイテムをすべてのワークシートの順序付けられたコレクションを表すように単一のアイテムに折りたたみます。 ブック内に複数個のワークシートが含まれており、ワークシートを名前で表示オプションを有効にしない場合、全てのワークシートが単一のワークシートとして扱われます！これにより任意の数のワークシートを一度に処理することができるようになりますが、そのためには処理するワークシート全てが同一の構造をしてい

	る必要があります。
挿入 ( )	現在選択されているワークシートの前に、新たなワークシートを挿入します。
追加 ( )	新たなワークシートを追加します。アイコンの右側にあるテキストフィールドに名前を入力してください。
削除 ( )	現在選択されているワークシートを削除します。
Excel ファイルから新規シートを取得	新しいワークシートが検索した新しいワークシート内のコンポーネントを取得します。この設定は、関連性が高く、入力Excel ファイル (Excel 2007+ コンポーネント設定 を参照)ファイルを持つソースコンポーネント内で有効化されます。

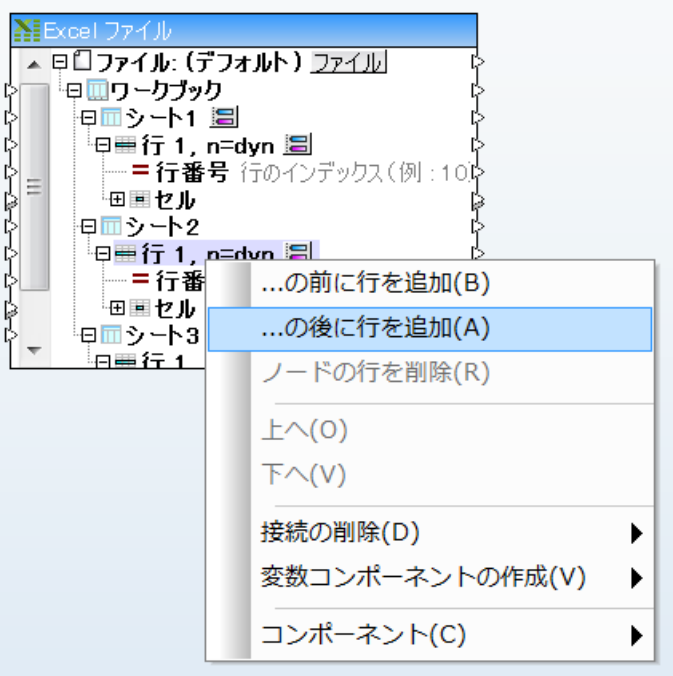
7.5.15.4 行の範囲の追加と移動

MapForce にワークシート内の特定の場所にあるセルの範囲に読み込み (または書き込み)を行うように命令することができます。以下の サンプルコンポーネントは、ワークブックのシート2で使用することのできる範囲について表示指しています。



デフォルトでは、行 1から開始する範囲を設定し、¥ n 行 (**n=dyn**) 反復することができます。必要な場合、しかしながらこれらと他の設定を変更することもできます (セルの範囲の選択を参照してください)。

同じワークシート内で必要な数量だけ行の範囲を作成することができ、必要でない行を削除することもできます。新規の行の範囲を追加するには、行 ( ) ノードを右クリックし、 **前に行を追加」** または **後に行を追加」** をそれぞれ選択します。



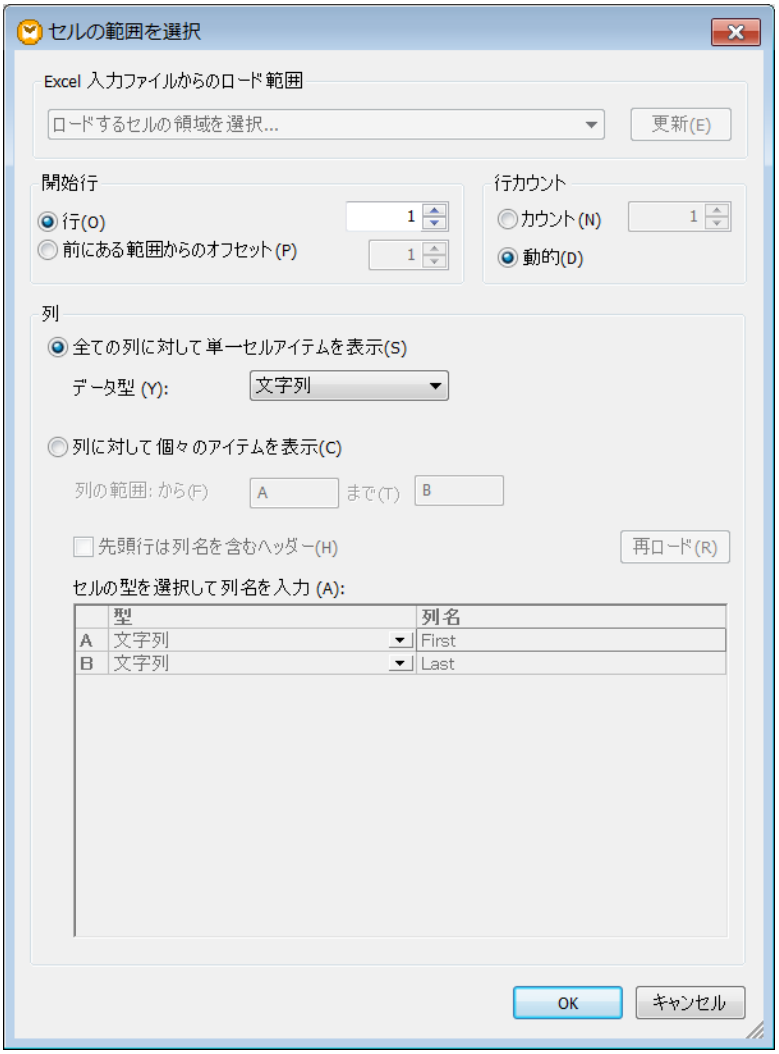
セルの範囲の追加

範囲を削除するには、右クリックし、**「行の削除」**を選択します。

コンポーネント内の範囲を上にまたは下に移動するには、右クリックし、**「上に移動」**(または**「下に移動」**)を選択します。

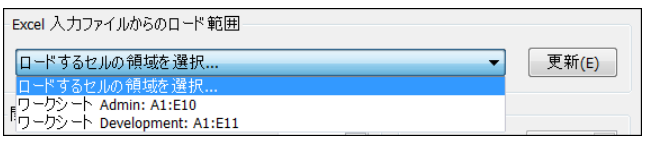
7.5.15.5 セルの範囲の選択

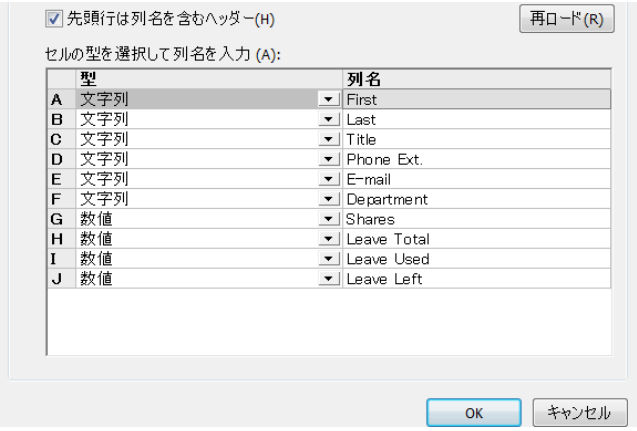
MapForce により(ワークブックから)読み込まれる、または(ワークブックに書き込まれるセルの範囲を"セルの範囲の選択"ダイアログボックス定義することができます。このダイアログボックスを開くには、コンポーネントのセルの範囲の横の ボタンクリックします。

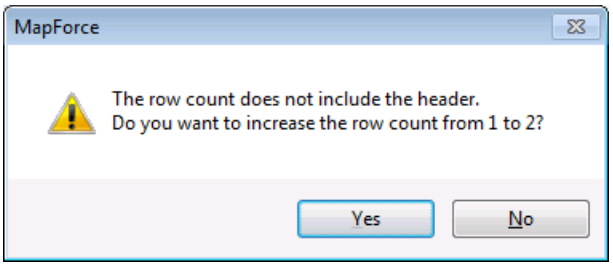


セルの範囲を選択 ダイアログボックス

使用することのできる設定は、以下のとおりです。

Excel 入力ファイルから範囲をロードする	**ワークブックから読み込む場合** このオプションを特定のワークシートの範囲、名前の付いた範囲、またはテーブルに対して使用します。  **ワークシートを名前順に表示する** オプションが無効化されている場合（ワークシートの追加と削除を参照して）、すべてのワークシートのデータがリストに表示されます。 **『更新』**ボタンは、入力 Excel ファイルからのセルの範囲を更新します。

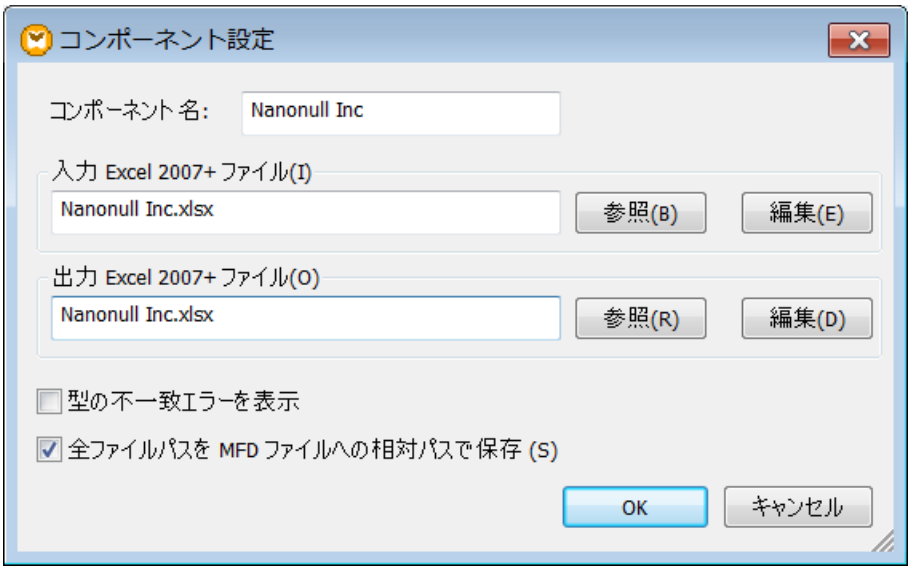
	MapForce では現在短形の範囲しかサポートされていないことに注意してください。 **ワークブックに書き込む場合** このオプションを使用することはできません。
先頭行	**行** オプションにより特定の範囲のためのデータの最初の行を定義することができます。 例： "5"を開始される行に入力すると、MapForce はワークブックの5番目の行を読み込み (または書き込み) ます。 **前にある範囲からのオフセット** オプションは同じワークシート内に他の範囲が存在する場合役に立ちます。 これはMapForce に現在の範囲の*N* 行を前に定義された範囲から移動するように命令します。最小のオフセットの値は 1です。
行カウント	**ワークブックから読み込む場合** **カウント**は、**先頭行** で定義された箇所から始まるデータを読み込む行数を定義します (前のオプションを参照してください)。 この値は、Excel の名前の付けられた範囲またはテーブルを選択すると自動的に作成されます。 **動的** オプションは、MapForce に**先頭行**から開始するソースデータ内のすべての行を読み込みます。 このオプションは、範囲がワークシートの最後に定義された範囲である場合のみ使用します。それ以外の場合、次の範囲はソースExcel ファイルからデータを選択しません。 **ワークブックに書き込む場合** **カウント**は、**先頭行**で定義されている場所から開始するデータが書き込まれる行の数を定義します。 前のオプションを参照してください)。 警告のないデータの残りは無視することができます オプション**動的** はMapForce に入力インスタンス内のすべての行を**先頭行**から書き込むように命令します。 **先頭行は列名を含むヘッダー** オプションを使用するヘッダー行を定義すると **カウント** はヘッダー行を数に数えません (**Excel_Company_to_XML.mfd** サンプルを参照) 
すべての行のための単一セルアイテムを表示する	下で表示されるように、すべてのアイテムを単一のマップすることのできるセルアイテムに折りたたみます。

	 **ワークブックから読み込む場合** 特定の行のすべてのセルから読み込む場合は、このオプションを使用します。例えば、**ExcelColumnsToRecords.mfd** と **ExcelWith2Dimensions.mfd** は **MapForceExamples** プロジェクト内で使用することのできるサンプルです。**OOXML Excel 2007+** フォルダーを参照してください。 **ワークブックに書き込む場合** 1つまたは複数のセルにデータを書き込む場合このオプションを使用します。例えば、 **Altova_Hierarchical_Excel.mfd** サンプルを参照してください。
列のために個別のアイテムを表示する	このオプションにより与えられた行範囲の個別の行にアクセスすることができます。 範囲またはテーブルという名前のワークシートの範囲を選択した場合、列名は自動的に作成されます。それ以外の場合、特定の列名を対応するアルファベット文字を**from** から**to** へのテキストボックス入力することにより選択することができます。 入力 Excel ファイル内の範囲が変更されると、変更と共にコンポーネントを更新するために **「再ロード」** をクリックします。 MapForce にその範囲の列ヘッダーとして最初の行を数えるように命令するには、**先頭行は列名を持つヘッダー** オプションを選択します。**行カウント**がセットされた状態でこのオプションを有効/無効にすると、行カウントの値を調整するか尋ねられます。このプロンプトは行カウントの数が 1 つ多く、または少なくなるのを防ぐためのものです。.  **行カウント** 設定に入力された行の数には、ヘッダー行が含まれないことに注意してください。

7.5.15.6 Excel 2007+ コンポーネント設定

マッピングエリアに "Excel 2007+" コンポーネントを追加すると、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定ダイアログボックスを以下の方法で開くことができます：

- **「コンポーネント」**メニューから**「プロパティ」**をクリックします。 (このメニューアイテムは、コンポーネントを選択すると有効化されます。)
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」**をクリックします。



コンポーネント設定ダイアログボックス

使用することのできる設定は、以下のとおりです。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名が自動的に生成され、インスタンスファイルをその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。 コンポーネント名はスペースとフルストップ文字を含むことが出来ます (例 : "ソース ファイル") (例 : "Orders.xlsx")。コンポーネント名スラッシュ、バックスラッシュ、コロン、二重引用符、行頭および行末スペースを含むことは出来ません。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください : • マッピングをFlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとWindows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
Excel 2007+ ファイルの入力	MapForce がデータを読み込むMicrosoft Excel ファイルを指定します。ソースコンポーネント内で、MapForce が Excel ワークシートの名前とカラムを読み込むために使用します。 ファイルのロケーションを変更するには、 **「参照」**をクリックして、新しいファイルを選択します。Microsoft Excel 内のファイルを編集するには、 **「編集」**をクリックします。

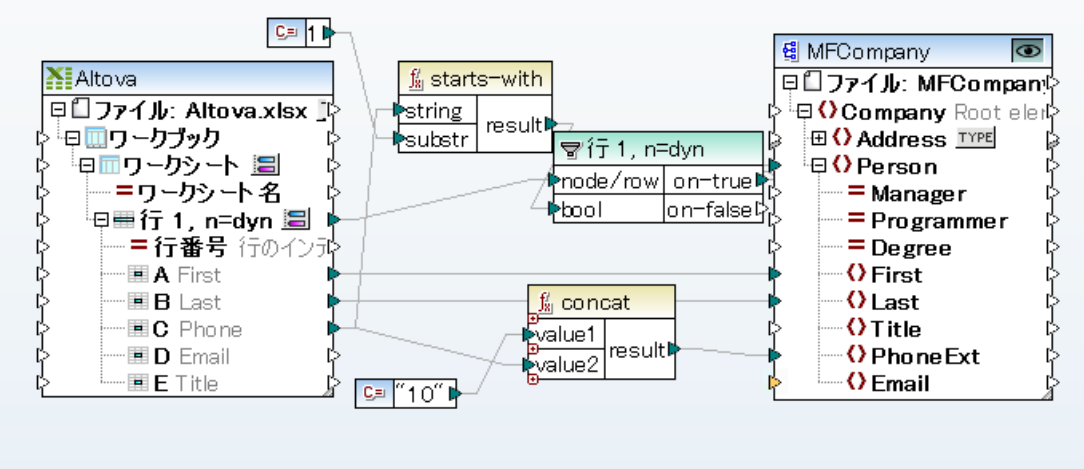
Excel 2007+ ファイルの出力	MapForce がデータを書き込む Microsoft Excel ファイルの名前/パスを指定します。このフィールドはターゲットコンポーネント内はとても重要です。 ファイルのロケーションを変更するには、**参照...** をクリックして、新しいファイルを選択します。Microsoft Excel 内のファイルを編集するには、**編集...** をクリックします。 出力 ファイルを指定しない場合、MapForce は、**xlsx-mapfore.xlsx** という名のファイルを生成します。デフォルトでは、このファイルは、異なるパスが構成されていない限り、と同じフォルダー内に生成されます。 マッピング変換が実行されると、出力 Excel ファイル内にある既存のデータは上書きされます。
型の不一致のエラーを表示する	コンポーネント内で宣言されたデータ型と入力 xlsx ファイル内のデータの間に不一致が生じると、メッセージウィンドウ内でエラーメッセージが用事されることを有効化します。 例：この設定が有効化されていると仮定し、*セルの範囲を選択するダイアログボックス* 内で、数値として列を宣言します。Excel ファイルがこの列にテキストデータを含む場合、この列を出力に接続し、出力を確認する場合にエラーメッセージが表示されます。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。以下も参照：コンポーネント上で相対パスを使用する。

7.5.15.7 例 Excel 2007+ を XML にマッピングする

以下に示す例は **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial\Excel-Mapping.mfd** にて確認することができます。最初の 2 つのマッピングがここでは取り上げられます。最初の 2 つのマッピングがここでは取り上げられます。

このマッピングの目的：

- Excel のブック内から、カラム C にある) PhoneExt が "1" から始まっている Person だけをフィルタリングします。
- "10" をオリジナルの番号に追加 連結して、該当する従業員 (Person) の名前、苗字とともに MFCompany XML ファイルへ渡します



Excel-mapping.mfd (サンプル 1)

マッピングは、以下のように構成されています：

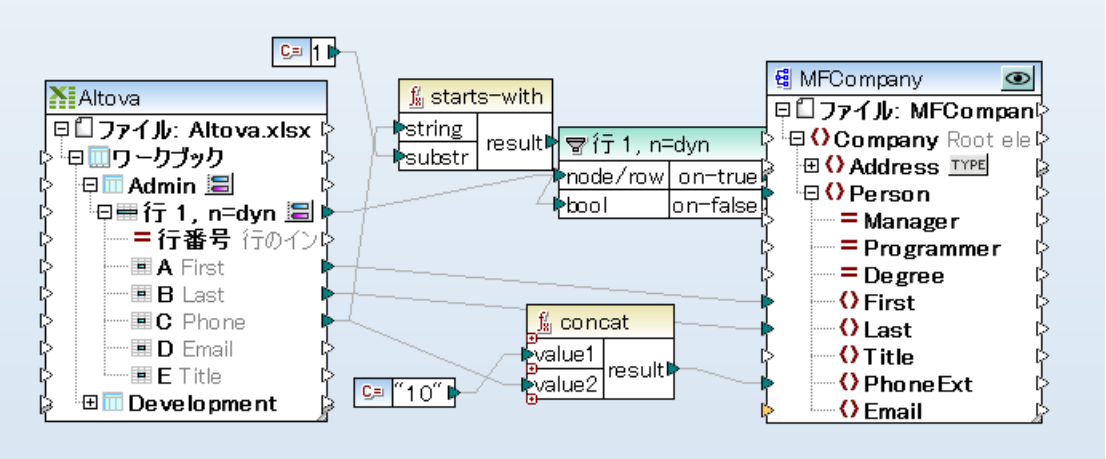
- **Altova.xlsx** はソースExcel 2007+ ワークブックです。カラムA ならびにB にて名前と苗字がそれぞれ与えられており、カラムC には電話の内線番号が入力されています。
- ブックの各ワークシートに対して **「各シートを名前で表示」**オプションは無効になっています。これは「ワークブック」アイテム以下にあるワークシートのタイトルが、「ワークシート」となっていることからも確認できます。
- **starts-with** 関数により、カラムC の内線番号が "1"から始まるかチェックされ、その結果が真 (true) である場合、そのレコードがフィルターコンポーネントから渡されます。
- **concat** 関数により、"10"という文字列が各内線番号の先頭に挿入され、PhoneExt アイテムへ渡されます。
- データの出力が行われる際には、ターゲットコンポーネントのMFCompany.xsd インスタンスファイルにフィルタリングされた従業員の情報が収められます。

4人の社員がそれぞれの詳細と共にXML ファイルへマッピングされます。

```
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>
```

2 番目のマッピングでは、**「各ワークシートを名前で表示」**オプションにより、ソースXLSX コンポーネントにある個々のワークシートが有効になった結果が表示されます。このマッピングは以下のように構成されています：

- **Admin** および**Development** ワークシートの両方がブックアイテム以下に表示されます。
- ターゲットコンポーネントへの接続は、**Admin** ワークシートからのみ行われています。



Excel-mapping.mfd (サンプル 2)

マッピングの結果では 2 人の従業員が XML ファイルへマッピングされます。

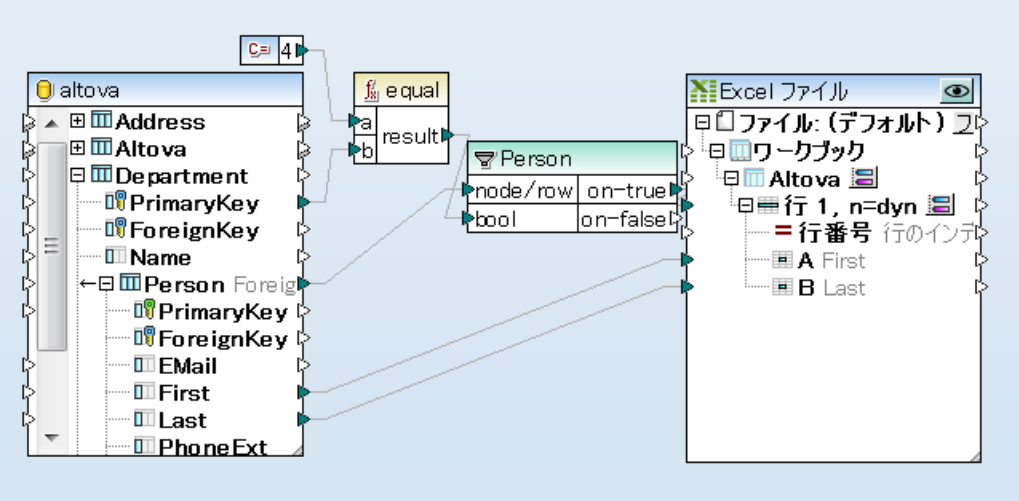
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
3    <Person>
4      <First>Steve</First>
5      <Last>Meier</Last>
6      <PhoneExt>10114</PhoneExt>
7    </Person>
8    <Person>
9      <First>Max</First>
10     <Last>Nafta</Last>
11     <PhoneExt>10122</PhoneExt>
12   </Person>
13 </Company>
```

7.5.15.8 例 :データベースデータを Excel 2007+ にマッピング

以下に示す例は **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーに収められている**Excel-mapping.mfd** にて確認することができます。 3 つあるうちの一番下のマッピングがここでは取り上げられます

マッピングの目的は、以下のとおりです :

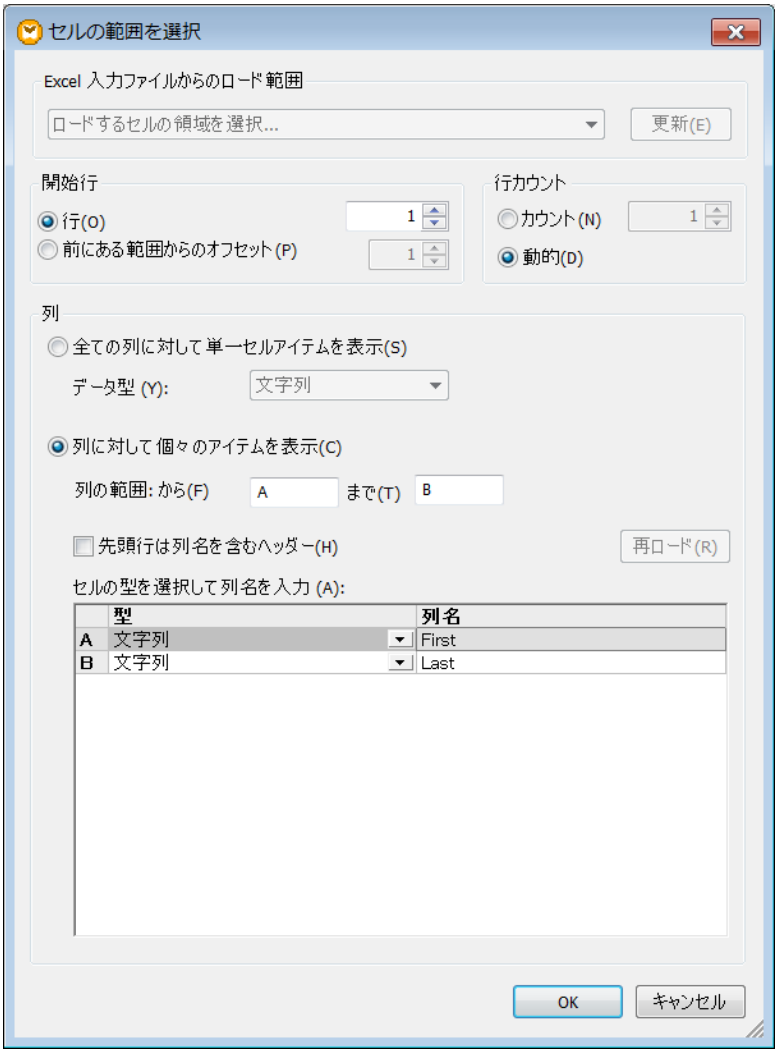
- altova データベース内にある Person データのうち、Department の主キーが 4 (IT 部門) になっている従業員 (Person) をフィルタリングします
- IT 部門の従業員データを、"空の" Excel ファイルテンプレートへ挿入します。



Excel-mapping.mfd (サンプル 3)

マッピングは、以下のように構成されています:

1. **挿入 | データベース** メニューコマンドを使用して、データベース「Altova」がマッピングエリアに **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーから追加されます。Microsoft Access データベースのためのウィザードに従います。
2. **挿入 | Excel 2007+ ファイル** メニューコマンドを使用して、Excel 2007+ が追加されます。サンプルファイルを与えるオプションはスキップされます。
3. ワークシート名の「Altova」は、シート1アイコン をクリックして、ワークシート名に「Altova」と入力することで定義することができます。
4. **行 1、n=dyn** 範囲は ボタンをクリックすることにより、構成されます。セルの範囲のオプションは以下のように定義されています:



5. その他のオプションは以下のように定義されています :

   - **PrimaryKey** の値が **定数** コンポーネントから与えられた "4" という値と `equal` 関数により比較されます。
   - **フィルター** コンポーネントは、ブールの値が true である場合 (すなわち、department 主キーが "4" の場合) に、値の受け渡しを行います。node/row パラメーターに接続されているソースコンポーネントの子アイテム/ノードのコンテンツがターゲットコンポーネントへ渡されます。
   - **on-true** アイテムは Excel ファイル。内の **行 1,n=dyn** アイテムに接続されています。

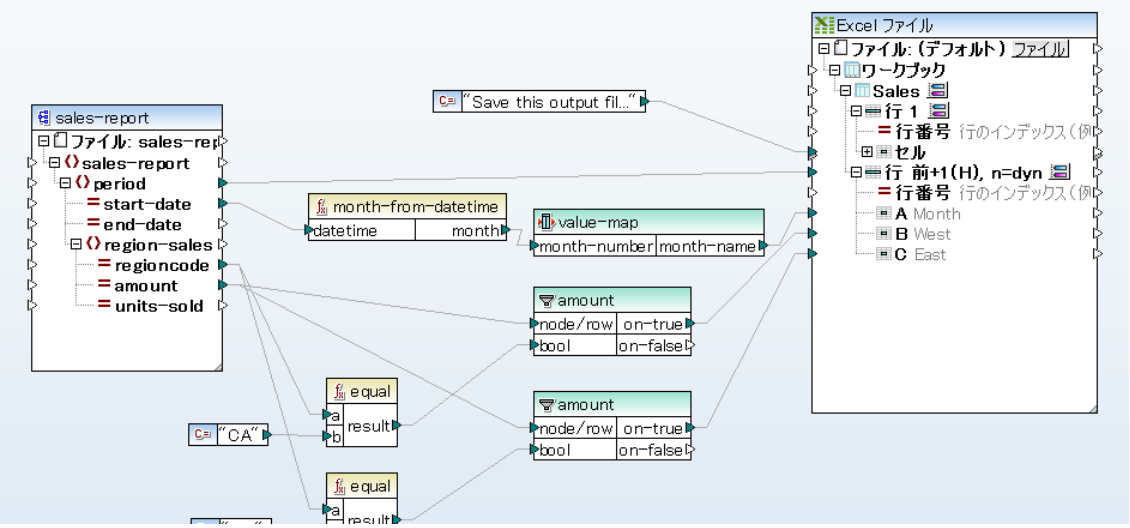
マッピングの結果は、IT 部門にいる4人の従業員名が Excel ブックに表示されます。

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

Altova

7.5.15.9 例 :定義済み Excel シートへデータを提供する

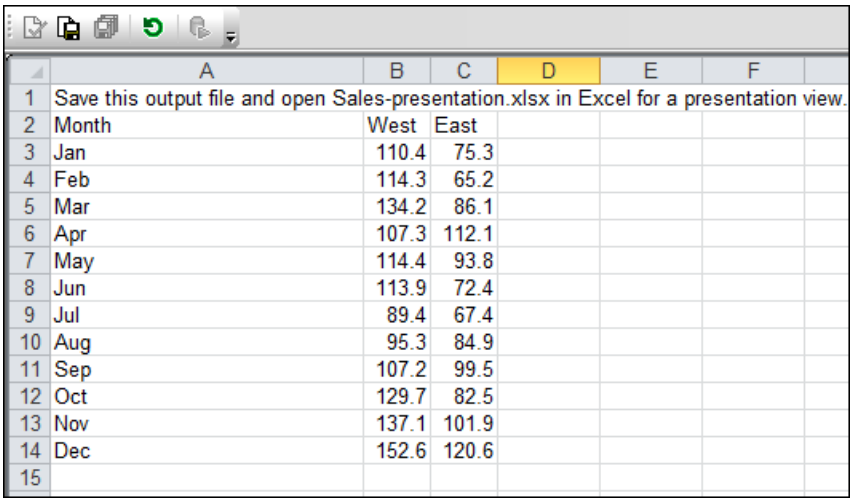
Excel シートの出力であるMapForce データは、のフォーマット済みのExcel ドキュメントのデータソースとして使用することができます。スクリーンショットは、 <Documents>\Altova\MapForce2017\MapForceExamples\ フォルダー内で使用することのできる**Sales_to_Excel.mfd** マッピングを表しています。



Sales_to_Excel.mfd

マッピングの結果をフォーマットされていないExcel データソースとして保存する:

1. 出力言語を選択するために、BUILT-IN アイコンクリックします (プログラミング言語 とコンパイルから選択することもできます)。
2. 「出力」ボタンをクリックして、MapForce 内の埋め込まれたExcel シートとしての結果を確認します。
3. 生成された出力をアイコン  をクリックして保存し、提案されたファイル名 "xlsx-mapforce.xlsx" を与え、 ...\MapForceExamples フォルダー内のワークブックに保存します。

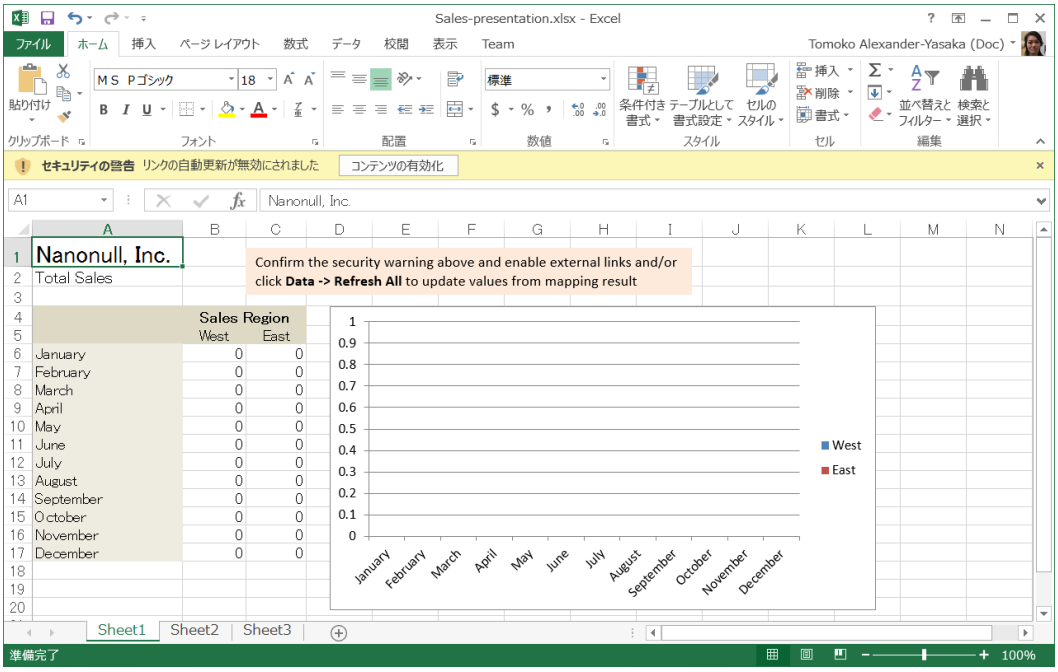


	A	B	C	D	E	F
1	Save this output file and open Sales-presentation.xlsx in Excel for a presentation view.					
2	Month	West	East			
3	Jan	110.4	75.3			
4	Feb	114.3	65.2			
5	Mar	134.2	86.1			
6	Apr	107.3	112.1			
7	May	114.4	93.8			
8	Jun	113.9	72.4			
9	Jul	89.4	67.4			
10	Aug	95.3	84.9			
11	Sep	107.2	99.5			
12	Oct	129.7	82.5			
13	Nov	137.1	101.9			
14	Dec	152.6	120.6			
15						

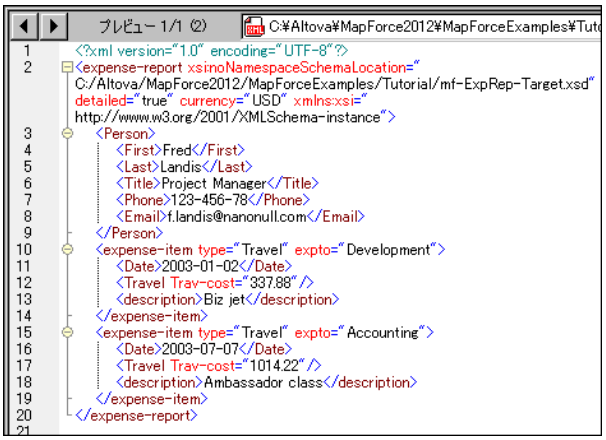
Excel テンプレートファイルのフォーマット

MapForce データを表示するために使用される新規のExcel ファイルが作成されると、MapForce 出力 ファイルデータを参照する式を追加し、シートにチャートを追加するなどします。\MapForceExamples フォルダー内の**Sales-presentation.xlsx** という名前のExcel ファイルはフォーマット済みのため、これらの処理を行う必要はありません。

1. **Sales-presentation.xlsx ファイル**をダブルクリックします。外部リンクが無効化されており、ソースデータにリンクされる必要があることを警告するメッセージが表示されます。現在のカーソルの位置、B6、では、ソースデータの範囲がxlsx-mapforce.xlsx ファイルの**Sales'!$B$3:$C$14** であることが表示されています。



2. ボタンクリックします。（または メニューオプション「データ」すべて更新」を選択します）外部リンクは有効化されており、基となるソースデータテーブルとグラフとして表示されています。



7.5.16 XBRL

Altova Web サイト XBRL ファイルのマッピング

XBRL (Extensible Business Reporting Language) は、ビジネスの情報の交換を可能にするオープンな国際標準です。XBRL は、政府、金融機関、ビジネス で金融および規制の報告のために使用されます。XBRL は、グローバルな有利を目的としないXBRL インターナショナル (https://www.xbrl.org) という協会により管理されています。

企業または機関は (例えば、XBRL 検証、 XBRL データの作成、新規XBRL タクソノミの作成、XBRL データを他のフォーマットにエクスポートするなど)XBRL に関して異なるニーズを持っている可能性があります。多種のAltova 製品によりXBRL に関連するタスクを達成することが可能になりました:

- XMLSpy は、新規タクソノミを編集または作成し、XSLT 変換ファイルをベースにしたXBRL レポートを生成します (XSLT 変換ファイルはStyleVision により作成することができます)
- StyleVision は、タクソノミスタイルシートテンプレートを作成し、XBRL レポートを生成します。
- MapForce は、XBRL インスタンスファイルからのデータをマップし、マッピング構造をデザインする際にXBRL タクソノミを使用することを可能にします。XBRL データをMapForce (Microsoft Excel 2007 と以降、データベース、または、CSV ファイルなど) がサポートするフォーマットからマップ または、フォーマットにマップすることができます。例えば、既存のXBRL インスタンス ドキュメントから特定のデータをフィルターすることができます。または、XBRL インスタンスをデータベースから抽出されたデータから作成することができます。
- *RaptorXML+XBRL Server* は、XBRL インスタンスとタクソノミを検証するクロスプラットフォームXBRL プロセッサー です。
- *AltovaのExcel のための仕掛品 (WIP) XBRL アドイン*により、XBRL 米国仕掛品 レポートを提出するビジネスは、Microsoft Excel

を使用してXBRL レポートデータを確認し、準備することができるようになりました。

MapForce 内のXBRL インスタンスファイル からデータをマップ、または、インスタンスファイルにデータをマップするために、基となるXBRL タクソノミが必要になります。タクソノミは、機関から提供される、または、XBRL データが提出されるエンティティである (通常 zip 形式の) ファイルアーカイブから構成されています。次に挙げられるタクソノミの例は、MapForce を使用したデータマッピングのために使用されています:

- 財務会計基準審議会 (http://www.fasb.org) により公開されているUS-GAAP (一般に認められた会計原則)
- IFRS (国際財務報告基準) (http://www.ifrs.org) により公開されているIFRS (国際財務報告基準)
- 欧州銀行監督局 (http://www.eba.europa.eu) により公開されているCOREP / FINREP (共通報告と財務報告)

適切なタクソノミプロバイダーからタクソノミを取得している限り、データマッピングを追加条件無しに作成することができます。しかしながら、デフォルトでは、US-GAAP タクソノミの最も適切なバージョンがMapForce に含まれており、検証が最適化されます。他のタクソノミは、ローカルにインストールされている場合以外、ロードと検証に時間がかかります。追加タクソノミ (または、古いタクソノミバージョン) をローカルにインストールするには、Altova コンポーネントダウンロードページでダウンロードすることのできるXBRL タクソノミインストーラー を使用してください (http://www.altova.com/download_components.html)。

MapForce は、次のXBRL に関連した標準をサポートします:

- XBRL 2.1
- XBRL Dimensions 1.0
- XBRL Table Linkbase 1.0 (次を参照してください:XBRL テーブルと作業する)
- XBRL Taxonomy Packages 1.0 (次を参照してください:XBRL タクソノミパッケージ)
- Transformation Registry 3 を含む インラインXBRL 1.0 と1.1 (データの読み込みのみに関する情報は、インラインXBRL からデータを読み込むを参照してください)。

変換

このドキュメントは以下の仕様により定義されているXBRL 用語について言及されています:

仕様	URL
XBRL 仕様 2.1	http://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html
XBRL ディメンション仕様 1.0	http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html
Table Linkbase 仕様 1.0	http://www.xbrl.org/Specification/table-linkbase/REC-2014-03-18/table-linkbase-REC-2014-03-18.html
XBRL タクソノミパッケージ 1.0	https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html
インラインXBRL	https://specifications.xbrl.org/spec-group-index-inline-xbrl.html

このドキュメントでは、仕様列により示された仕様について参照しています。さらに、§ 文字は仕様内の特定のセクションの番号を表すために使用されています。例 :XBRL Table Linkbase 1.0 仕様のセクション 5.4 への参照は、以下の通りです :

テーブル (Table Linkbase 仕様 1.0、§5.1) は、XBRL コンポーネント内の ▦ アイコンにより示されています。

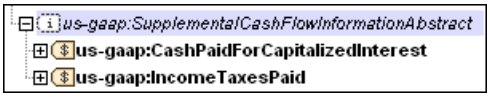
7.5.16.1 XBRL ファイルをマッピングコンポーネントとして追加する

XBRL ドキュメントにデータをマップ または、XBRL ドキュメントにデータをマップする前に、XBRL ドキュメントは、マッピングエリアにマッピングコンポーネントとして追加される必要があります。XBRL ドキュメントをマッピング コンポーネントとして追加するには以下の2つの方法があります :

- フラット XML ドキュメントとして
- 階層的構造を持つXBRL ドキュメントとして

最初のオプションは、を、ビジネスロジックコンセプトまたは特定のXBRL 階層構造無しに、フラットXML ファイルとして挿入します。このようなコンポーネントは、次元の処理や、自動的なXBRL コンテキストの生成が存在しないシンプルなマッピングにとても役にたちます。

2番目のオプションは、XBRL タクソノミを挿入し、そのコンテンツをXBRL に適する階層的な方法で表示します。この場合、MapForce は、Discoverable タクソノミSet (DTS) のレファレンスを解決し、自動的に派生したXBRL 構造をビルドします。これは、XBRL マッピング コンポーネントが、XBRL 特有のアイテム (XBRL ファイル内の実際のノード または DTS アイテムから派生した 抽象的 "ノード) を含むことを意味します 例 :次の階層的構造は、`xlink:from` と `xlink:to` プレゼンテーションアークから派生し、プレゼンテーション リンクベースとして振る舞います。("リンクベース"という用語は、XLink 仕様内で定義されている用語と同じ意味があります (http://www.w3.org/TR/xlink11/ を参照してください)。具体的には、リンクまたはリンクデータベースのコレクションを示します。 XLink リンクベースは タクソノミ内で定義された追加情報を与えるために、XBRL に拡張されます。)



このサンプルは、プレーンXML では無くマッピングオペレーションに適切な、概念的な方法でデータを表示するために、MapForce のタクソノミの基本的な解決方法を説明しています。かどのようにタクソノミを解決するかを説明する基本的な表示です。プレゼンテーション リンクベースに加え、MapForce は、Table Linkbase を含む他のリンクベースから作成さ

れた構造も表示することができます（XBRL テーブルと作業するを参照してください）。

コンベンショナルなグラフィックの表示を含むXBRL コンポーネントに表示されることのできるアイテムに関する詳細は、XBRL コンポーネントアイテムについてを参照してください。

XBRL ドキュメントをフラット XML として開く

- **挿入**」メニューから「**XML/スキーマファイル**」をクリックします。

自動的なコンテキストの処理を使用して XBRL ドキュメントを開く

1. **挿入**」メニューから「**XBRL ドキュメントの挿入**」をクリックします。
2. タクソノミファイル（*xsd）またはXBRL インスタンスファイル（*xbrl または *xml）を開きます。タクソノミを選択すると、有効なインスタンスファイルを選択するためのダイアログが表示されます。
3. このXBRL コンポーネントをソースインスタンスとして使用する場合、**参照**」をクリックして、XBRL インスタンスファイルを選択します。

> XBRL タクソノミを開くと、ダイアログボックスが開き、XBRL コンポーネント内で表示される構造ビューを選択するようにプロンプトされます。どの構造ビューを選択するのか不定の場合は、デフォルトのオプションを選択してください。デフォルトでは、MapForce は自動的にロードされたDiscoverable Taxonomy Set (DTS) からデータを分析してオプションを選択します。構造ビューを後に必要に応じて変更することができます（構造ビューの選択も参照）。

7.5.16.2 インライン XBRL からデータを読み込む

XBRL と作業する機関の一部は、XBRL の代替（または、追加）として、インラインXBRL フォーマットの情報を提供、または送信の受付、をする場合があります。インラインXBRL、または、iXBRL は、マシンにより読み取ることのできるHTML ドキュメントに埋め込まれたXBRL コンテンツを参照します。これにより、をWeb ブラウザー内で人間が判読できるXBRL データを表示することができます。iXBRL に関連する仕様のリストに関しては、次を参照してください：https://specifications.xbrl.org/spec-group-index-inline-xbrl.html。

MapForce では、（htm、html、または、xhtml ファイルなどの）インラインXBRL を含むファイルをマッピングにデータソースとして追加することができます。すなわち、MapForce は、サポートされるターゲットのフォーマットにマッピングするために、インラインXBRL インスタンスファイルからデータを読み取ることができます。インラインXBRL インスタンスへのデータの書き込みは、サポートされていないことに注意してください。

必要条件

- マッピング変換言語はBUILT-IN に設定されている必要があります（次を参照してください：変換言語の選択）。他の言語はサポートされていません。BUILT-IN 変換言語を使用して、マッピングをMapForce で直接確認するか、サーバー実行可能ファイルにコンパイルして、MapForce Server で実行して確認することができます（次を参照してください：MapForce Server 実行可能ファイルにマッピングをコンパイルする）。
- インラインXBRL ファイルのタクソノミは、URL アドレス、または、ローカルファイルへの参照である場合があります。後者の場合、マッピングがMapForce Server により作動する場合、以下の追加条件に関して注意してください：
  - タクソノミファイルは、mfx（マッピング実行）ファイルと同じディレクトリに存在する必要があります。
  - インラインXBRL ファイルは、相対的パスを使用するタクソノミファイルを指す必要があります、例：`<link:schemaRef xlink:type="simple" xlink:href="taxonomy.xsd" />`。
  - コンポーネント設定 ダイアログボックスは、相対的パスを使用するタクソノミファイルを指す必要があります（例えば、"taxonomy.xsd"）（次を参照：XBRL コンポーネント設定）。

- ソースXBRL ファイルは、単一のXBRL インスタンスを含む必要があり、複数のインスタンスは、サポートされていません。

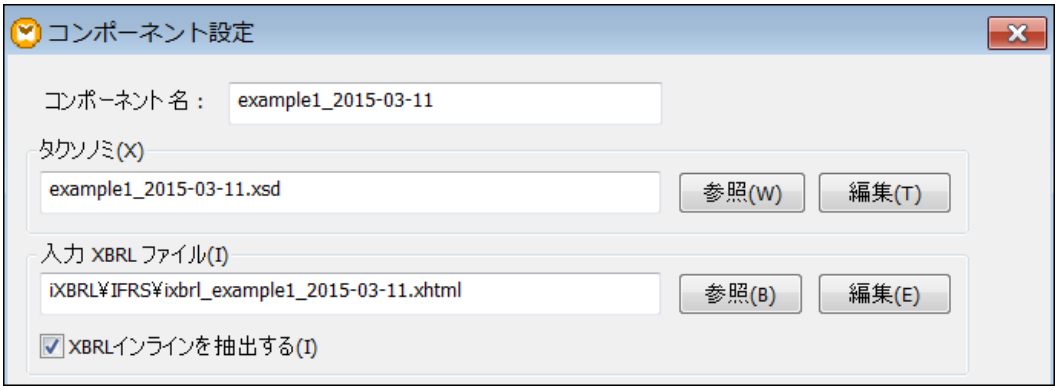
インライン XBRL を含むインスタンスファイルをマッピングに追加する

1. **「挿入」**メニューから、**「XBRL ドキュメント」**をクリックします。
2. 次のうちの1つを行います：
   a. XBRL タクソノミがある場合、XBRL タクソノミ（通常、 xsd 拡張子を持つファイル）を参照します。これは、インラインXBRL インスタンス ファイルにとって妥当なタクソノミである必要があります。URL をタクソノミとして指定します。これをおこなうには、ブラウザーダイアログボックス上の **「URL に切り替える」**ボタンをクリックします（URL からコンポーネントを追加するも参照してください）.
   b. XBRL タクソノミファイル またはURL が無い場合、インラインXBRL インスタンス ファイルを代わりに参照してください。この場合、MapForce は、自動的に対応するXBRL タクソノミを決定しロードします。
3. インスタンスを提供するようにプロンプトされると、インラインXBRL を含むHTML ファイルを参照します（通常、 html、 htm または xhtml 拡張子を持つファイル）。
4. **「入力 XBRL ファイル」**の横のインラインXBRLを含むファイルを参照します（通常の場合、ファイルには html または xhtml 拡張子が付いています）。
5. **「OK」**をクリックします。

この時点では、インラインXBRL コンテンツの構成は、コンポーネント内で使用することができます。検証エラーが存在する場合、メッセージウィンドウに表示されます。それ以外の場合、通常のXBRL コンポーネントと同様にインラインXBRL コンポーネントからXML、テキスト、データベース などのターゲット構造に接続を描きます。

「インライン XBRL の抽出」チェックボックス

XBRL コンポーネント設定 ダイアログボックス上の **「インラインXBRL の抽出」**チェックボックスが選択されると、 html または xhtml ファイルが、インラインXBRL として扱われる入力として指定されます。この結果、MapForce は、インラインXBRL コンテンツを、入力ファイルから抽出し、ダイアログボックスが閉じられると検証します。このチェックボックスは、マッピングにインラインXBRL インスタンス ファイルを追加すると通常自動的に選択されます。オプションが有効化されているかを確認するには、マッピング上のXBRL コンポーネント ヘッダーをダブルクリックします（または、代わりに、XBRL コンポーネント ヘッダーを右クリックして、**「プロパティ」**を選択します）。



7.5.16.3 XBRL タクソノミ パッケージ

XBRL タクソノミ パッケージは、オフラインのXBRL タクソノミのコピーを含むZIP アーカイブです。XBRL タクソノミパッケージの利点は（タクソノミ プロバイダから提供されている場合）スキーマなどのXBRL リソースがインターネットから抽出される

代わりにローカルの (オフラインの) タクソノミファイルから抽出されます。 XBRL タクソノミパッケージの構造とフォーマットは タクソノミパッケージ 1.0 勧告で説明されています (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html)。

使用中の XBRL タクソノミが上記の勧告に準拠する XBRL タクソノミパッケージのダウンロードを提供する場合、このようなパッケージを MapForce に追加することができます。これは XBRL 検証と処理に影響を与えます (XBRL スキーマレファレンス等の特定の URL がタクソノミパッケージに含まれるローカル ファイルにリダイレクトされるなどと同じ意味です)。タクソノミパッケージは、XBRL タクソノミの全てのエントリポイントを MapForce から直接確認することができ、XBRL 検証のパフォーマンスを向上します。

メモ: 次の Altova デスクトップ アプリケーションは XBRL タクソノミパッケージの追加をサポートします :XMLSpy、MapForce、および StyleVision。XBRL タクソノミパッケージは、これらのアプリケーション全てに共通のものをリストします。1 つのアプリケーション内のものを編集すると、編集されたリストが他のアプリケーションでも表示されます (XBRL 検証と処理が影響を受けます)。

作動方法

各 XBRL タクソノミパッケージは内部の カタログファイル (**catalog.xml**) を含みます。このカタログファイルは、ローカルリソースにリダイレクトされる (マップし直される) URI を定義します。カタログファイルは、制限された XML カタログ仕様のサブセットに準拠します (https://www.oasis-open.org/committees/download.php/14809/xml-catalogs.html)。同様のメカニズムが MapForce にも搭載されており、をローカルリソースに URI をマップし直すことができます (次を参照してください : カタログファイル)。

1 つ または、複数の XBRL タクソノミパッケージを追加した後、(下の手順を参照してください)、MapForce がどのように XBRL 検証のために リソースを解決するかを影響します。例えば、XBRL インスタンスをマッピングに追加すると、スキーマが、インターネットから抽出される代わりに、ローカル タクソノミパッケージからロードされます。 (タクソノミパッケージがアクティブでその **catalog.xml** ファイルがスキーマ URI をローカル リソースにマップし直すと想定します)。この結果、これは以降のすべての XBRL 検証と処理に影響を与えます。

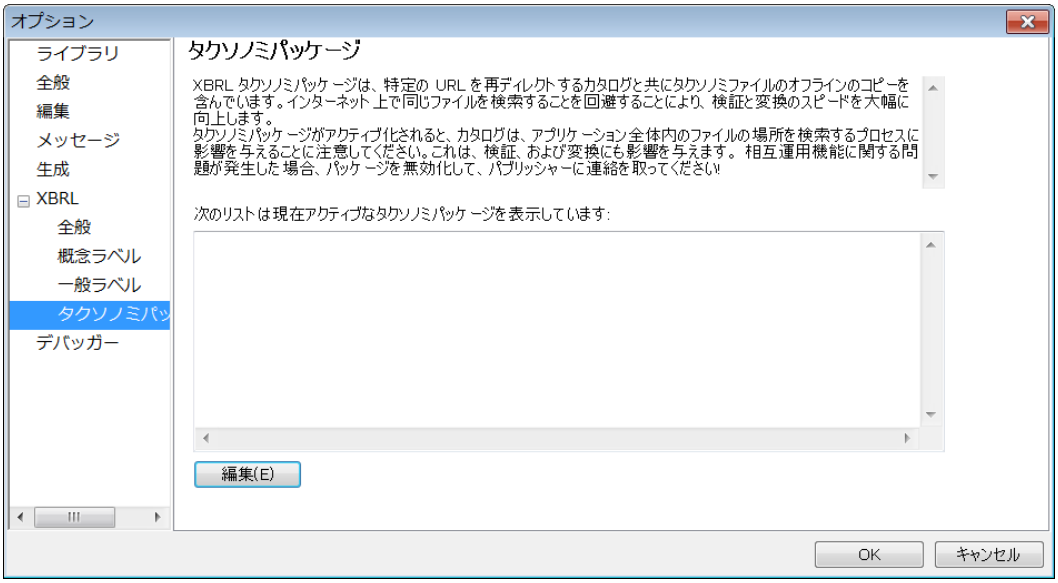
注意 : パッケージ カタログは、互換性の無いリソースにリダイレクトされる可能性があります
アクティブなパッケージのカタログファイルにより指されるリソースは、そのリソースを必要とするすべての MapForce オペレーションで使用されます。このようなリソースのサンプルは、XML 検証 と XBRL 検証に使用される XML スキーマです。パッケージのカタログファイルによりロケートされるオフラインのリソースは、使用中の既存の環境と互換性が無いために エラーが発生する場合があります。この場合、タクソノミパッケージを無効化して、エラー情報をパッケージの作成者に伝えてください。

タクソノミパッケージの追加

1. XBRL タクソノミパッケージ (zip アーカイブ) をプロバイダからダウンロードし、頻繁に変更される予定のないローカルディレクトリに保存します。ダウンロードされた アーカイブを解凍しないでください。

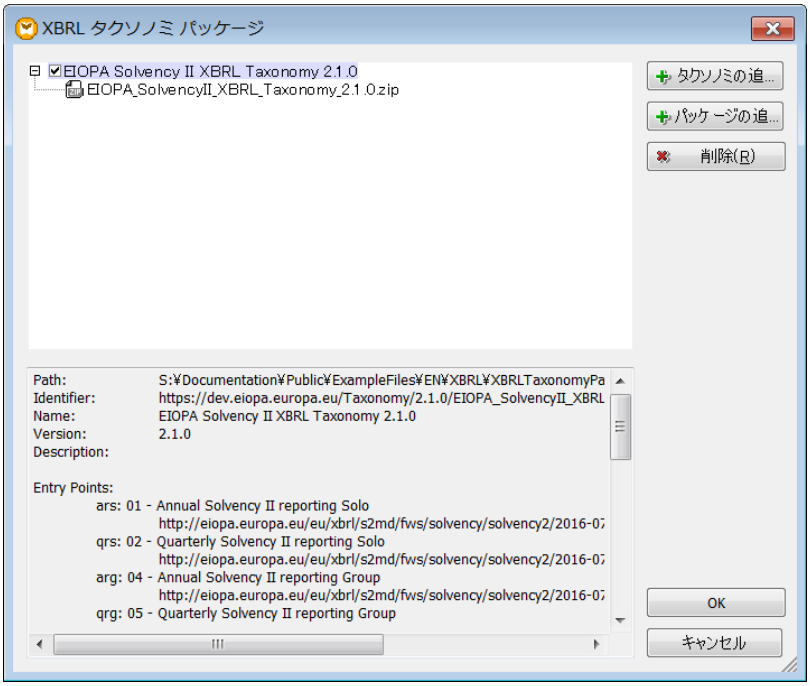
メモ: ダウンロードされた zip アーカイブは、タクソノミパッケージ 1.0 勧告に準拠する必要があります (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html)。

2. **「ツール」** メニューから **「オプション」| XBRL | タクソノミパッケージ」** をクリックします。



3. **「編集」**をクリックします。
4. **「タクソノミの追加」**をクリックして、.zip タクソノミパッケージを参照します。複数のパッケージを選択する場合は、**「Shift」**キーを押します。

追加されたXBRL タクソノミパッケージはダイアログボックス内に2つのレベルで表示されます。最初のレベルはタクソノミを表し、2番目のレベルは、そのタクソノミのパッケージを表します。XBRL タクソノミパッケージ ダイアログ内でタクソノミパッケージが選択されると、(オフラインの場所も含まれる)詳細がダイアログの下のペインに表示されます。



タクソノミの左横のチェックボックスのエントリは、タクソノミがアクティブであるか表示します。デフォルトでは、新規に追加されたタクソノミがアクティブです。タクソノミを無効化するには、チェックボックスのチェックを解除してください。無効化は、タクソノミの2つのバージョン間を切り替える際に問題が発生すると、役に立つ機能です。

追加パッケージを既存のタクソノミに追加する場合、タクソノミを選択し、入力して、**「パッケージの追加」**をクリックします。追加されたパッケージは、そのタクソノミの2番目のレベルに表示されます。パッケージを削除するには、選択して**「削除」**をクリックします（これは、ディスクから実際のタクソノミを削除しないことに注意してください）。

7.5.16.4 XBRL コンポーネントアイテムについて

以下のテーブルは、MapForce の一部である XBRL コンポーネントアイテム（ノード）の概要です。ノードの一部は、対応する構造ビューが選択されている場合のみ使用することができます（構造ビューの選択を参照してください）。例：の場合、プレゼンテーション定義リンクベースからのみデータを表示することが選択されている場合、テーブルリンクベースに特有のノードは、表示されません。

コンポーネントアイテム	アイコン	説明
抽象アイテム		抽象アイテムは関連するファクトを編成するために使用されます。表示リンクベース、または定義リンクベース内にあるドメインメンバーネットワークにて定義することができます。これらが XBRL インスタンスファイルにて実際に記述されることはありませんが、直感的な方法でファクトのグループ化を行うことができます。
ブレークダウン		テーブルリンクベース仕様 1.0、§5.4 内で定義されているとおり。
コンテキスト		コンテキストノードは、関連する全てのビジネスファクトを収めるコンテナーです。ハイパーキューブ内部にあるコンテキストノードにより、コンテキスト要素や次元が自動的に管理されます。XBRL ルート要素の子である xbrli:context ノードは、次元を手動で管理する際に使用されます。
次元		次元（XBRL Dimensions 仕様 1.0, §2.5）はビジネスファクトのためのコンテキスト情報を構造するために使用されます。次元はハイパーキューブ内のタクソノミにより定義されます。
明示的なメンバー		明示的なメンバーは QName 値の次列挙により定義される次元メンバーです（コンポーネント内の次元の表示も参照）。
明示的なメンバーの値		明示的なメンバーの値
ファクト		ファクトとは（XBRL 仕様 2.1, §1.4）XBRL アイテムの値のことです。以下の型があります： • 通貨アイテム（ ） • 文字列アイテム（ ） • 数値アイテム（ ） • 一般アイテム（ ） • 株式アイテム（ ）
フットノート		フットノートでは、概念に対してテキスト情報を新たに加えることができます。
ハイパーキューブ		ハイパーキューブでは（XBRL Dimensions 仕様 1.0, §2.2）、定義リンクベースや表示リンクベースから得られた情報を使って、階層構造の次元、概念、そして関連する XBRL 概念が構成されます。XBRL ハイパーキューブと作業するを参照してください。
ルート要素		xbrli:xbrl 要素が XBRL コンポーネントのインスタンスルート要素となります。

コンポーネントアイテム	アイコン	説明
ルールノード		テーブルリンクベース仕様 1.0、§6.6 内で定義されているとおり。
構造ノード		テーブルリンクベース仕様 1.0、§5.5 内で定義されているとおり。
テーブル		テーブルリンクベース仕様 1.0、§5.1 内で定義されているとおり。MapForce は、テーブル構造を階層に表示します（XBRL テーブルと作業するを参照してください）。
テーブルセット		テーブルリンクベース仕様 1.0、§5.2 内で定義されているとおり。
タプル		Tuples (XBRL 仕様 2.1,§1.4) タプルはファクトやその他のタプルをメンバーとして含んだ複合型の要素です
型指定されたメンバー		型指定されたメンバーは XML スキーマ要素 により定義されます（ http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html#sec-typed-dimensions を参照してください）。
ユニット		ユニット要素 xbrli:unit にはXBRL アイテムが参照するユニットが含まれます。データのマッピングを行う際には、xbrl:unit 要素に対して値を定義する必要があります。例えば、米ドルに対して UnitID = usd ならびに Measure = iso4217:USD という値が使用されます。XBRL デフォルトと作業するを参照してください。
表示		表示は、XBRL タクソノミの定義とプレゼンテーションの拡張されたリンクロールを示します。

アイコン は、ジェネリックで、 XBRL ルート要素を含む多種のノードの型に使用されます。

更に、以下の *グラフィカルユーザーインターフェイス要素* は、コンテキストで必要とされる場合、XBRL コンポーネントで表示されます。

GUI 要素	アイコン	説明
追加情報		このアイコンは、エラーなどのメッセージに関連したアイテムと共に表示されます。このアイコンをクリックして、メッセージペインに追加情報を表示することができます。
エラー		このアイコンは、エラーメッセージを示します。
コンテキスト メニューの表示		このアイコンは、このセクションで使用することのできる追加コンテキストメニューオプションと共に表示されます。
未知		このアイコンは、無効な名前空間への参照、または参照されたアイテムがタクソノミ内に存在しない場合、MapForce により解決できないアイテムと共に表示されます。

7.5.16.5 構造ビューの選択

XBRL ドキュメントをマッピングエリアに追加する際、コンポーネント上でXBRL 構造ビューが表示されるように、（ XBRL ファイルをマッピングコンポーネントとして追加するを参照してください ）を選択することができます。または後に選択することもできます。使用することのできる構造ビューは、以下のとおりです。

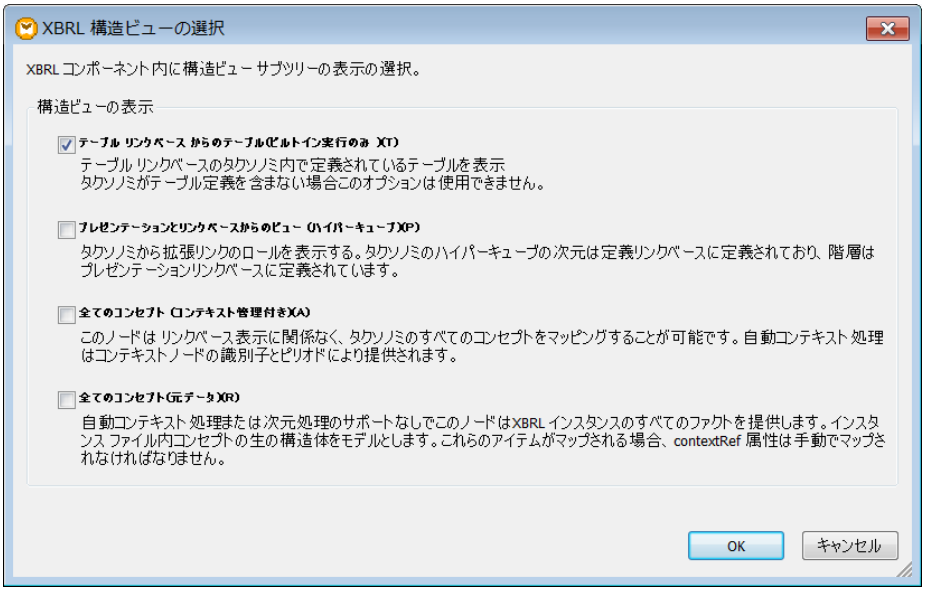
構造の表示	説明
テーブルリンクベースからのテーブル	タクソノミのテーブルリンクベース内で定義されたテーブルを表示します。タクソノミがテーブル定義を含まない場合、このオプションは無効化されます。 このビューは任意です。続行するには、このチェックボックスを選択するか、または **「プレゼンテーションおよび定義リンクベースからのビュー」** チェックボックスを選択します。
プレゼンテーションおよび定義リンクベースからのビュー	プレゼンテーション定義リンクベースからのデータを表示します。プレゼンテーションリンクベースはハイパーキューブと次元を持つ拡張されたリンクロールを含む階層構造と定義リンクベースを含みます。 このビューは任意です。続行するには、このチェックボックスを選択するか、または **「テーブルリンクベースからのテーブル」** チェックボックスを選択します。
全てのコンセプト (コンテキストの管理付き)	階層構造内の追加ノード"全てのコンセプト"を表示します。2つのデフォルトのディメンションの**識別子**と**ピリオド**を使用したハイパーキューブ内でレポートされるかに関わらず、このノードは、タクソノミのすべてのコンセプトのマッピングを有効化する、ハイパーキューブ"無次元"を含みます。自動的なコンテキストの処理は、タクソノミのすべてのXBRL 概念を子として含む、**コンテキスト**ノードにより提供されます。抽象アイテムは、表示されていません。 このビューは任意で表示されます。
全てのコンセプト (未フォーマット)	自動的なコンテキストの処理、または、次元の処理のサポートを持たないのすべてのXBRL インスタンスのファクトへのアクセスを与える"全てのコンセプト (未フォーマット)"を表示します。これは、インスタンスファイル内の概念のフォーマットされていないXML構造 をモデルとしていて、これらのアイテムをマッピングする場合、`contextRef` 属性を手動でマップしなければなりません。 このビューは任意で表示されます。

メモ

- XBRL コンポーネントからデータを読み込む場合、 (すなわち、XBRL コンポーネントがソースコンポーネントの場合)、構造ビューの組み合わせを選択することができます。
- XBRL コンポーネントに書き込む場合、 (すなわち、iXBRL コンポーネントがターゲットコンポーネントの場合) **「テーブルリンクベースからのテーブル」** は、他のビューとの組み合わせに使用することはできません。
- **「テーブルリンクベースからのテーブル」** ビューは"ビルドイン"が変換言語として選択設定されている必要があります (変換言語の選択も参照)。

XBRL コンポーネント内での構造ビューの表示の選択

1. XBRL コンポーネントのルート要素から **「コンテキストメニューの表示」** ( ) ボタンクリックして、**構造 ビュー** を選択します。



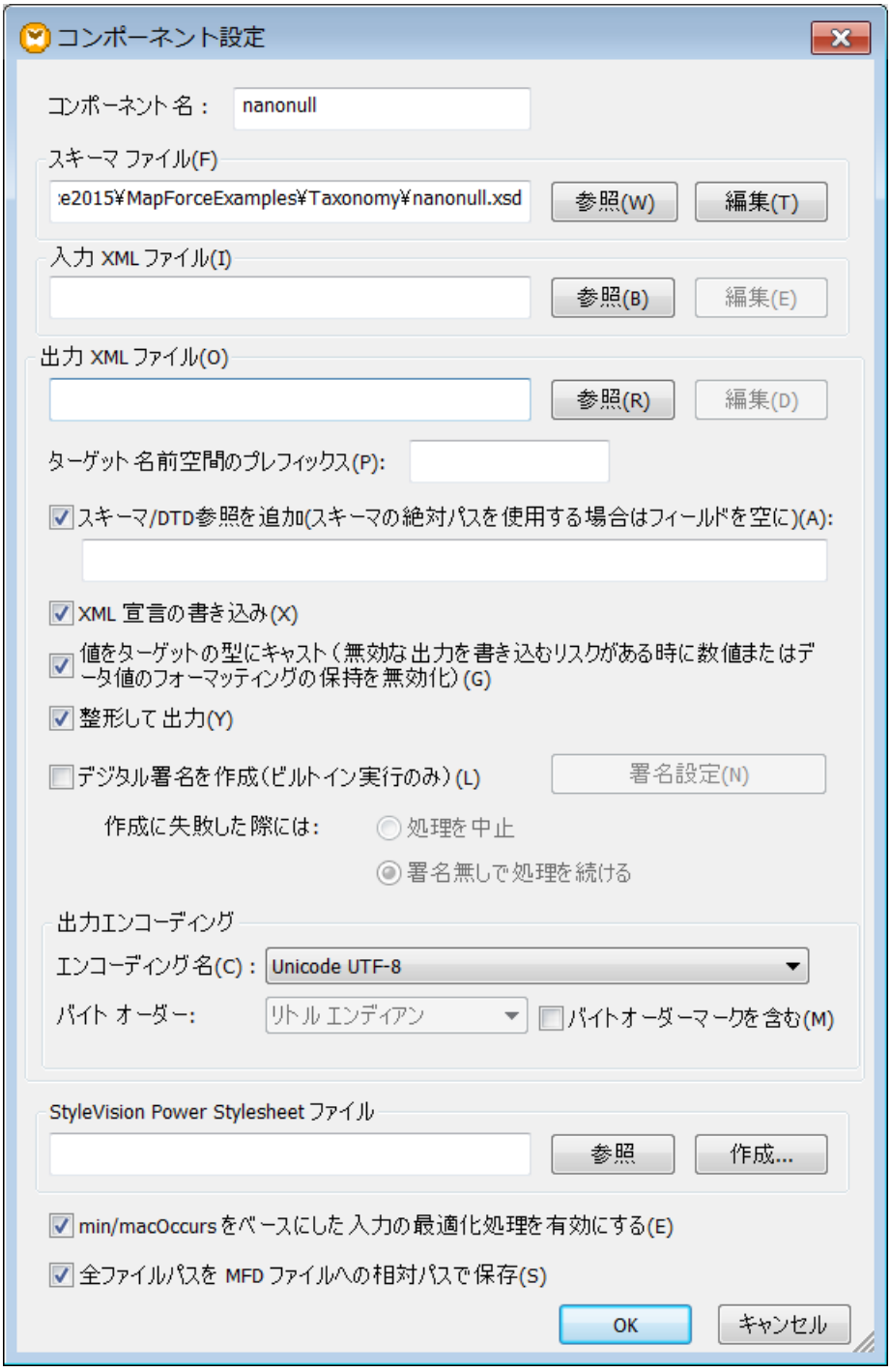
2. XBRL コンポーネント内で表示される必要がある、1つまたは複数の構造ビューを選択し、「OK」をクリックします。

7.5.16.6 XBRL コンポーネント設定

XBRL ドキュメントをマッピングエリアに追加すると(XBRL ファイルをマッピングコンポーネントとして追加するを参照してください)コンポーネントに適用することのできる設定をコンポーネント設定ダイアログボックスから構成することができます。

コンポーネント設定ダイアログボックスの開き方

- XBRL コンポーネントをダブルクリックします。



コンポーネント設定ダイアログボックスには以下の設定が含まれます。

タクソノミ	*メインのタクソノミファイルのファイル名とパスを指定します。* *ファイルのロケーションを変更するには、* **参照」***をクリックして、新しいファイルを選択します。*

	XBRL エディター (例 :XMLSpy) 内のファイルを編集するには、**「編集」**をクリックします。
入力 XBRL ファイル	現在選択されている XBRL コンポーネントのための入力 XBRL インスタンスのファイル名を指定します。このフィールドは XBRL コンポーネントを挿入して、XBRL インスタンスファイルを割り当てると自動的に入力されます。 ファイルのロケーションを変更するには、**「参照」**をクリックして、新しいファイルを選択します。 XBRL エディター (例 :XMLSpy) 内のファイルを編集するには、**「編集」**をクリックします。
出力 XBRL ファイル	コンポーネントがマッピングターゲットとして使用されている場合、XBRL ターゲットインスタンスファイルが保存される場所のファイル名とパスを指定します。 入力 XBRL インスタンスファイルを割り当てると、入力 XBRL ファイルフィールドからのエントリが自動的にこのフィールドにコピーされます。入力 XBRL インスタンスファイルをコンポーネントに割り当てない場合は、このフィールドはタクソノミファイルのファイル名とパスと拡張子 "xml" を含みます。 ファイルのロケーションを変更するには、**「参照」**をクリックして、新しいファイルを選択します。 XBRL エディター (例 :XMLSpy) 内のファイルを編集するには、**「編集」**をクリックします。
タクソノミスキーマレファレンス	MFD ファイルに相対した参照された関連したタクソノミスキーマファイルのパスです。このフィールドを検証のために異なるタクソノミを指定するために使用します。タクソノミ参照は、`link:schemaRef` 要素の `href` 属性内に書き込まれます。 `<link:schemaRef xlink:type="simple" xlink:href="..\..\nanonull.xsd"/>`
ターゲット型に値をキャストする	ターゲット XML スキーマ型がマッピングする際に使用されるかを定義することを許可します。またすべてのデータがターゲットコンポーネントにマップされ**文字列**の値として処理されるかを定義することができます。デフォルトでは、この設定 が有効化されています。
再フォーマットされた出力	構成された外観を与えるために、出力 XBRL ドキュメントを再フォーマットします。各子ノードは、単一タブ文字づつそれぞれの親のオフセット。
デジタル署名の作成	XBRL 出力インスタンスファイルにデジタル署名を追加することを許可します。 "ビルドイン" を変換言語として選択している場合、デジタル署名を追加することができます (デジタル署名も参照)。
エンコード	出力インスタンスファイルの次の設定を指定することを許可します : • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。

StyleVision Power スタイルシートファイル	このオプションは、Altova StyleVision スタイルシートファイルを選択または作成することを許可します。このようなファイルは XBRL インスタンスファイルからのデータをレポートのために HTML、RTF などの適切なフォーマットで出力することを有効化します。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定 は、以下のファイルに影響を与えます : • XBRL タクソノミファイル • XBRL 入力ファイル • XBRL 出力ファイル • StyleVision スタイルシートファイル 以下も参照 : コンポーネント上で相対パスを使用する。

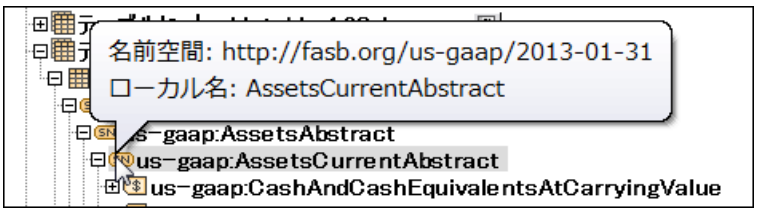
7.5.16.7 XBRL 基本設定

MapForce では、XBRL 特有の設定を以下のように構成することができます :

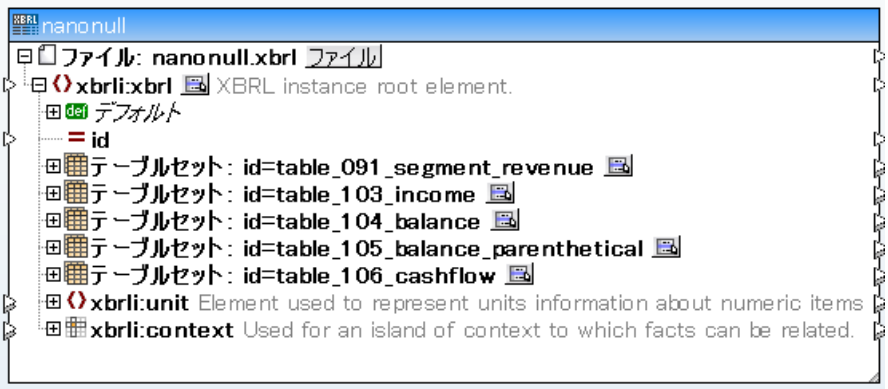
- コンポーネント アイテムに関する追加情報を確認する (ヒントと注釈の有効化を参照してください)。
- 一般的な (アプリケーション全体)XBRL 設定を構成する (全般 XBRL オプションを参照してください)。

ヒントと注釈の有効化

ヒントが有効化されていると、各コンポーネントアイテムについての追加情報を、マウスのカーソルをポイントすることにより、以下のように確認することができます。



(存在する場合)注釈テキストをオンとオフに切り替えることができます。有効化されている場合、注釈は、コンポーネント内でアイテムの右側に表示されます。 例 : 下のスクリーンショットでは、`xbrli:xbrl` ルート要素の注釈は、は "XBRL インスタンス ルート要素 " です。



ヒントを有効化、または、無効化するには、以下を行います：

- **表示**」メニューから**「ヒントの表示」**をクリックします。
- **「ヒントの表示」**（ ）ツールバーボタンをクリックします。

注釈を有効化、または、無効化するには、以下を行います：

- **表示**」メニューから**「注釈の表示」**をクリックします。
- **「注釈の表示」**（ ）ツールバーボタンをクリックします。

一般 XBRL オプション

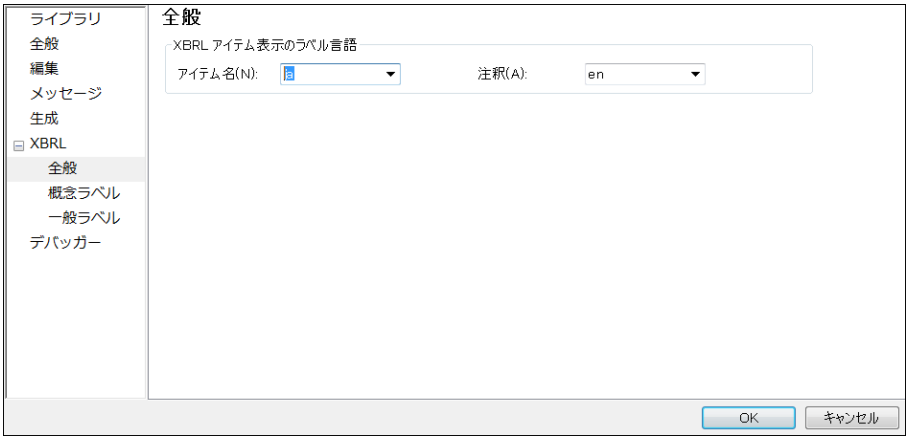
MapForce は、以下の全般的な（アプリケーション全体の）XBRL 設定を構成することができます：

- XBRL アイテムのラベル言語と注釈を設定します。
- XBRL アイテム名のための優先されるラベルロールを設定します。
- XBRL アイテムのための注釈のラベルロールの特定の型設定します。
- カスタム XBRL タクソノミパッケージを追加します（次を参照：XBRL タクソノミパッケージ）。

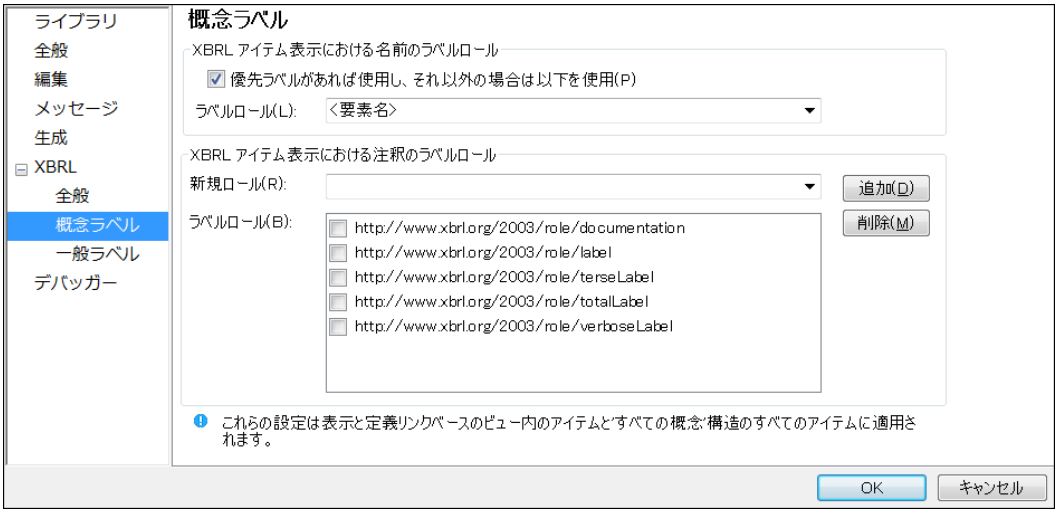
XBRL セクションの「オプション」ダイアログボックスから上の設定すべてを構成することができます。

全般 XBRL オプションの変更

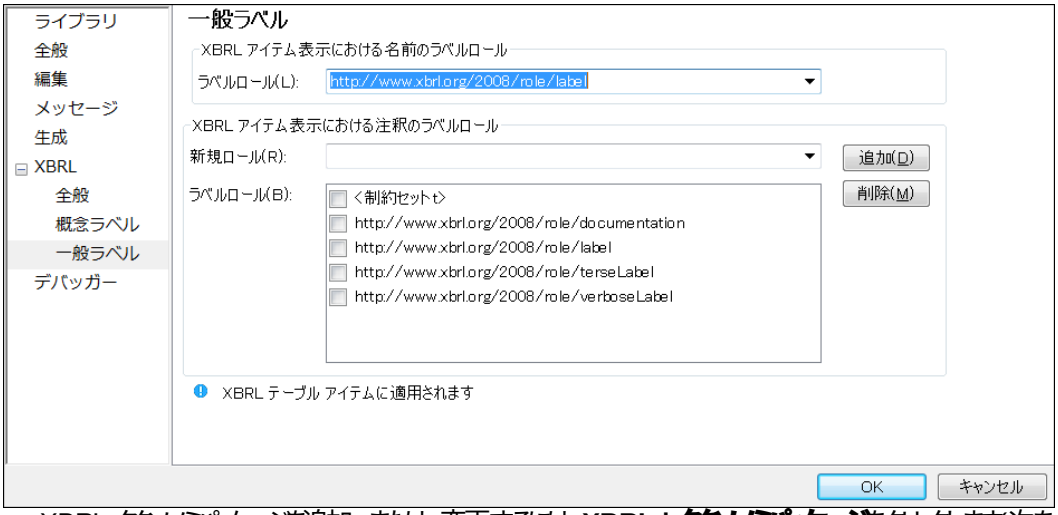
1. **表示**」メニューから**「XBRL 表示オプション」**をクリックします。
2. 以下を行います：
   - XBRL アイテムのラベル言語と注釈を変更するには、**XBRL | 全般**をクリックします。



- 以下の構成ビュー内のアイテムに適用することのできる優先順位を有するラベルロールと注釈を変更するには、**XBRL | 概念ラベル**:
  - *レゼンテーション定義リンクベースからのビュー*
  - *全てのコンセプト（コンテキストの管理付き）*
  - *全てのコンセプト（未フォーマット）*



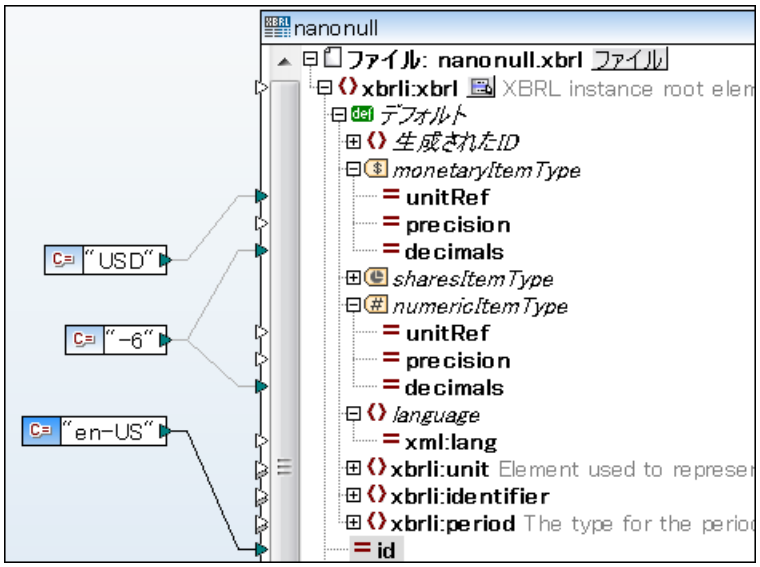
- テーブルリンクベースビュー内のアイテムのために適用することのできる優先順位を有するラベルロールと注釈を変更するには、**XBRL | 一般ラベル**をクリックします。



o XBRL *タクソノミパッケージ*を追加、または、変更するには、**XBRL | *タクソノミパッケージ***をクリックします(次を参照：XBRL *タクソノミパッケージ*)。

7.5.16.8 XBRL デフォルトと作業する

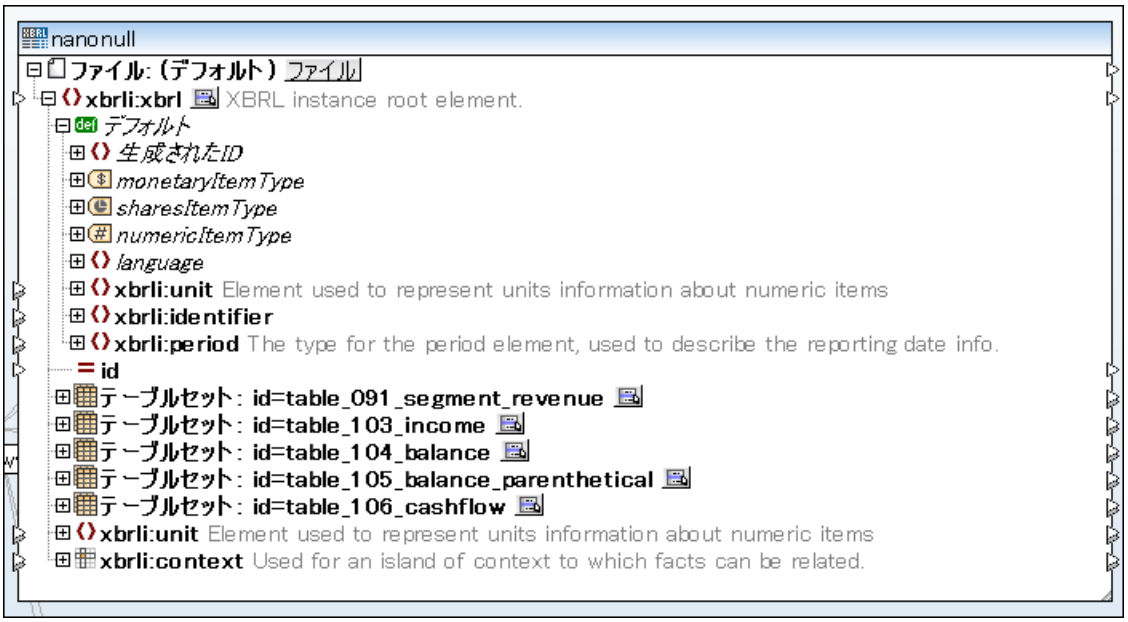
デフォルトは、XBRL コンポーネントの属性全てに対して明示的にマッピング接続することなく、属性の値を割り当てることのできる強力な方法です。 例： 以下のスクリーンショットに表示されるように定数ユニット識別子をデフォルトの階層内の`monetaryItemType` アイテムの属性 `unitRef` に割り当てると、入力が明示的に他の値にマップされている箇所を除いて、各金融 XBRL アイテムユニット識別子にデフォルトを割り当てます。下のスクリーンショットも、フットノートの言語を定義する`xml:lang` 属性のためのデフォルトの値の使用を表示しています。



デフォルトでは、 'デフォルト" ノードは、新規の XBRL コンポーネント内で表示されています。しかしながら、 'デフォルト" ノードを(XBRL ドキュメント全体のために)ルートレベルで、または、XBRL コンポーネントの全ての階層レベルで表示することもできます。 XBRL 構造内の異なったレベルで定義することができるため、異なるサブツリーは異なるデフォルトの値を持つことができます。

XBRL コンポーネントがプレゼンテーション/定義 リンクベースの代わりにテーブルリンクベースビューを使用する場合、テーブルリ

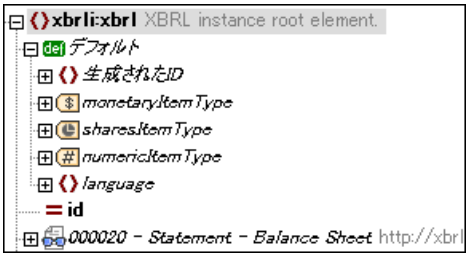
ンクベースから、アスペクト値のデフォルトをマップすることができます。下のスクリーンショットでは、xbrli:unit, xbrli:identifier とxbrli:period 要素は、マッピング to XBRL テーブルにマッピングする際に使用することのできるデフォルトのアスペクト値です。



特定のアイテムのための"デフォルト"ノードを表示する

1. デフォルトのユニットを表示するアイテムを右クリックし、 **「XBRL | デフォルトの表示」**を選択します (または、ノードのために使用できる場合は **「コンテキストメニューの表示」**( ) ボタンクリックして、 **デフォルトの表示**) をクリックします。

   これは、複数のアイテムの型のためのデフォルトの値に接続することのできるデフォルトのアイテムを挿入します。

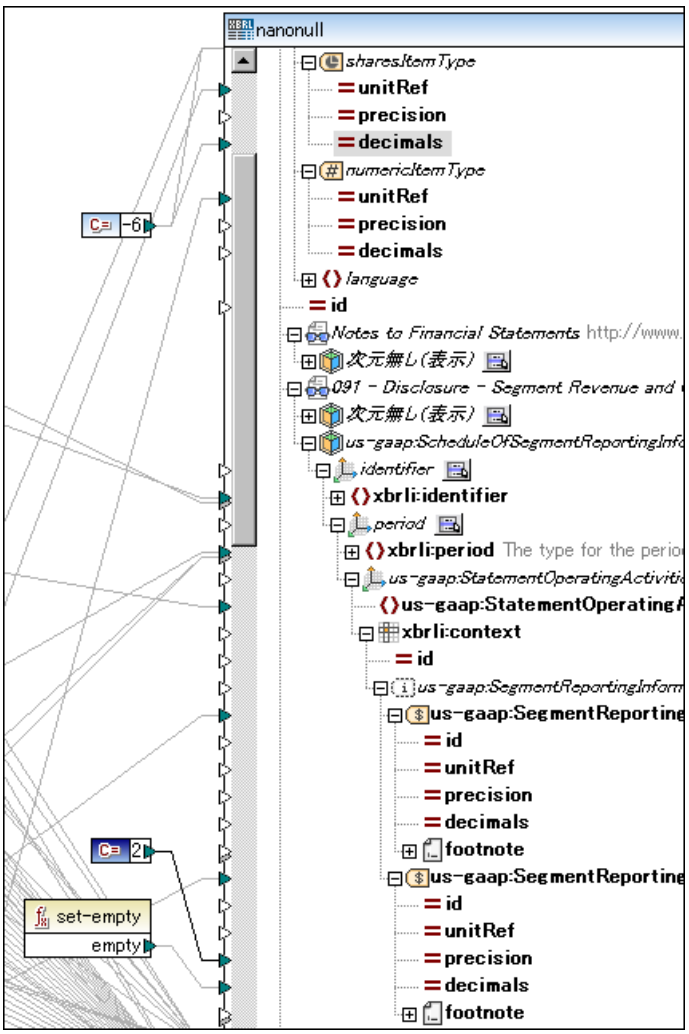


デフォルトの値を置き換える、または無効化する

デフォルトの値が一部のコンセプト属性、例えば、**decimals** のために定義されている場合、関数 **set-empty** を使用して、それぞれのコンセプトのためにこの設定を削除することが可能です。

下のスクリーンショットでは、**set-empty** 関数 "は、 "デフォルトの値 "-6"を金融アイテム、"Segment Reporting Information, Revenue" のために無効化します。

このアイテムは、 "2"と共に精度 属性にマップされます。他のアイテム、"Segment Reporting Information, Operating Income" は、 "-6"のデフォルトの小数値と共にレポートされます。



コンテキストの扱い方

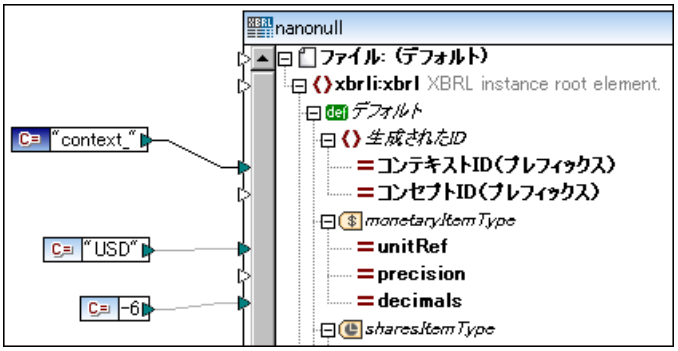
XBRL コンポーネント内の階層構造は、自動的なコンテキストの処理を許可します。XBRL 入力*インスタンス*内の `xbrli:context` の生成は、関連した概念をレポートする際、自動的に行われます。

XBRL *インスタンス*内の特定のコンテキストの属性 `id` の値は、XBRL 概念内の属性 `contextRef` の値です。MapForce は自動的に出力*インスタンス*内に作成された全てのコンテキストに番号を与えます。

"生成されたID "の下のノード "context id (prefix)"にテキストを*プレフィックス*として割り当てることによりカスタマイズが可能になります。

例 :定数値 "**context**_" をデフォルトの*プレフィックス*としてマッピングすると、値 "context_1"、 "context_2"、 "context_3" などの値を持つ連番のコンテキストID を出力*インスタンス*内に作成します。

デフォルトの値が定義されていない場合、MapForce は*プレフィックス* "**ctx**_"を持つコンテキストID を作成します。



出力 XBRL インスタンスが **フットノート** を含む場合、関連した概念は、自動的に生成されたフットノートのリンクをリンクするために概念 ID を持つ必要があります。これらの属性は自動的に生成されます。アイテム、"**concept id (prefix)**" は、このようなプレフィックスを決定するためにはっぷされることができます。

XBRL コンポーネント内でプレフィックスがマップされない場合、MapForce は、プレフィックス "**fact_**" を持つ全てのコンセプト ID を作成します。

7.5.16.9 XBRL ハイパーキューブとの作業

ハイパーキューブアイテムにより、自動コンテキスト処理が有効になります。XBRL ハイパーキューブには以下のような種類があります：

- タクソノミで定義されたもの。例：Statement [Table]
- MapForce により生成されたもので、**表示** リンクベースから派生したデフォルトの次元である identifier と period を簡略化して表示します。例：Dimensionless (presentation)
- **全てのコンセプトノード** 内で、MapForce により生成されたもの。例：Dimensionless

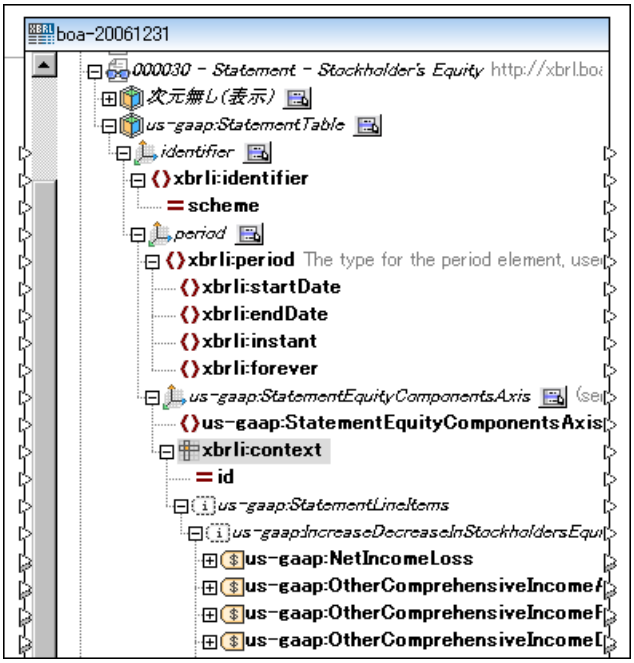
ハイパーキューブには、identifier と period という 2 つのデフォルト次元が含まれており、各コンテキストにあるこれら 2 つの要素に対して簡単に読み書きを行うことができるようになります。更に、タクソノミ内で定義された次元が、コンテキスト要素の xbrli:segment ならびに xbrli:scenario に対して自動的に関連付けられます。

ハイパーキューブを表す **「次元なし 表示)」** では両方の次元が使用されます。コンテキストノード内にて階層構造で表示されるコンセプトの順序は、表示リンクベースから得られます。

「次元なし」 ハイパーキューブアイテムでも両方のデフォルト次元が使用されますが、階層構造のコンセプトの順番は無く、タクソノミ内で定義された全てのコンセプトの元リストだけが表示されます。

その他のハイパーキューブは全てタクソノミ内で定義されており、タクソノミのラベルリンクベース内で定義された名前に従い表示されます。

ハイパーキューブとその次元（または軸）には、それぞれ小さなアイコン ic-dim-icon が表示されており、コンポーネント内にある各次元の表示を定義することができます。以下にあるイメージは、タクソノミファイルを開いた際のデフォルト状態を示します。



メモ:報告可能なコンセプトを含む全てのハイパーキューブがMapForce では表示されます。関連するハイパーキューブ次元がドメインを持たない場合、その次元はXBRL コンポーネントにて表示されません。

コンポーネント内で次元を表示する

XBRL 内にある次元アイテムは、インスタンス内部にて**明示的**または**型付**の値を参照します。各次元アイテムの注釈により、コンテキスト要素のxbrli:segment またはxbrli:scenario にて報告されるかを確認することができます。

型付次元により、XML スキーマ型の要素が表示されます。これらの値は直接マッピングされます。

XBRL タクソノミ内の**明示的**次元には特定のドメインから得られたxs:QName 型の値が与えられます。XBRL ドメインメンバー値と、XBRL ドメインアイテムそのものの値によりこの値は構成されます。明示的な次元は、マッピング、ならびにXBRL へマッピングしているその他のコンポーネント構造間における必須要件に合わせて 2 つのモードで表示することができます。

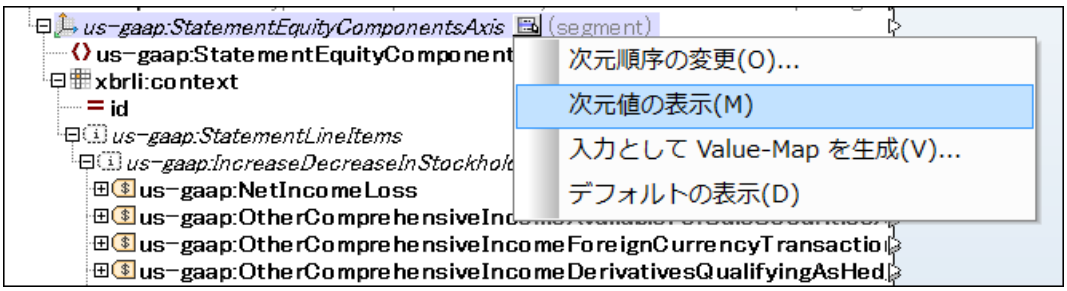
初期段階では、明示的な次元が**単一**の子ノードとともに表示され、この子へ直接マッピングすることができます (例:"Statement, Equity Components [Axis]")。

この方法は、データベースフィールドや、Excel テーブルのカラムといったソースデータ内にあるフィールドから、次元の値が派生するような場合にXBRL ターゲットコンポーネントに対して使用することができます。ソースデータには通常必要なQName データ型が含まれていないため、Value-Map 関数を使用してそれらを自動的に作成することができます (ハイパーキューブ次元のためにValue-Maps を生成するを参照)。

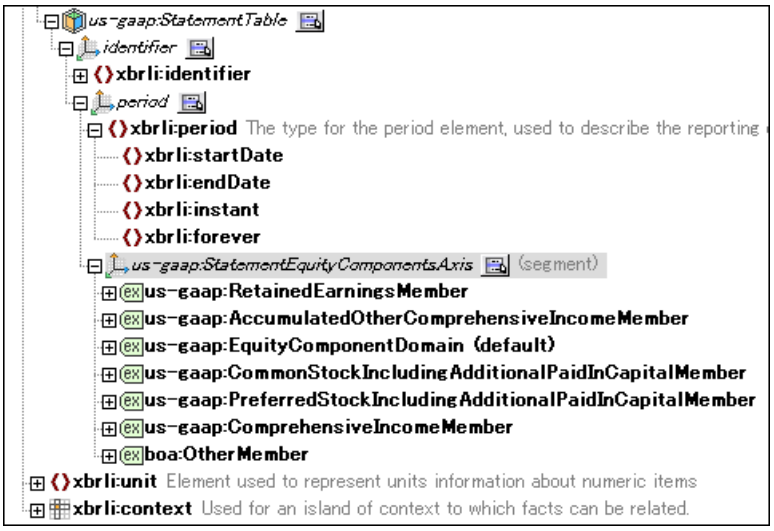
各次元メンバーに関連したファクトに対して異なるマッピングを行うため、次元ドメインにある個々の値に対して独立したノードを表示することができます。

コンポーネント内にある次元値を表示する:

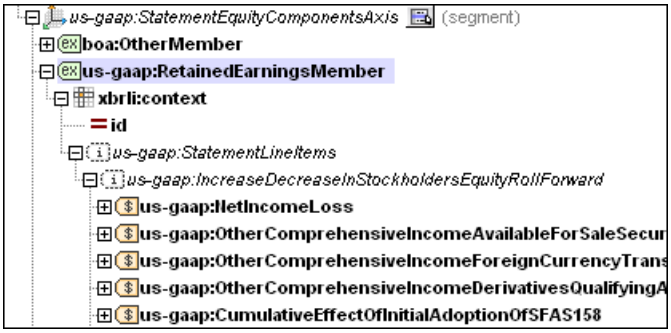
1. 値を表示する次元のアイコン をクリックして、「次元値の表示」を選択します。



この操作により、次元名以下のアイテムが表示されます。次元の**ドメイン**ならびに**メンバー**アイテムが表示され、それぞれ緑色のアイコンで表示されます。これらは全てドメインの明示的なメンバーで、アイテムアイコンには "ex" プレフィックスが表示されます。



明示的な各メンバーには、同じサブ構造が含まれており、個々に対して異なるマッピングを行うことが可能です。



コンセプトの出力に対してマッピングを行うと、これらの値が、適切な次元値を持った関連するコンテキスト要素に対して使用されます。例えば、"Statement, Equity Components [Axis]" 次元において "Comprehensive Income [Member]" 次元値を含むコンテキストに対してのみ、インスタンス内にある "Net Income (Loss)" という値がマッピングされます。フィルタリングを行う必要はありません。

XBRL 出力インスタンスを記述する際には、コンテキスト内部における適切な次元値の自動生成がサポートされます。例えば、必須報告アイテムの "Net Income (Loss)" に対して、コンテキストノードの xbrli:context が、コンテキスト要素 (xbrli:segment) 内部にて "Comprehensive Income [Member]" という値を含む明示的な要素の "Statement, Equity Components [Axis]" を取得します。

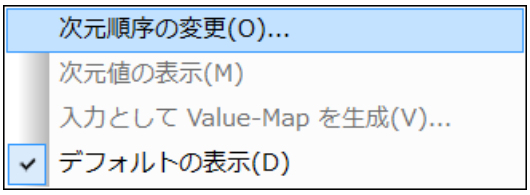
次元の順序を変更する

初期段階では、ネストされた子ノードというかたちでハイパーキューブにある全ての次元が表示され、階層構造が形成されます。ハイパーキューブ内部における次元の階層構造順序を変更することで、(非 XBRL の) マッピングに対応することができます。

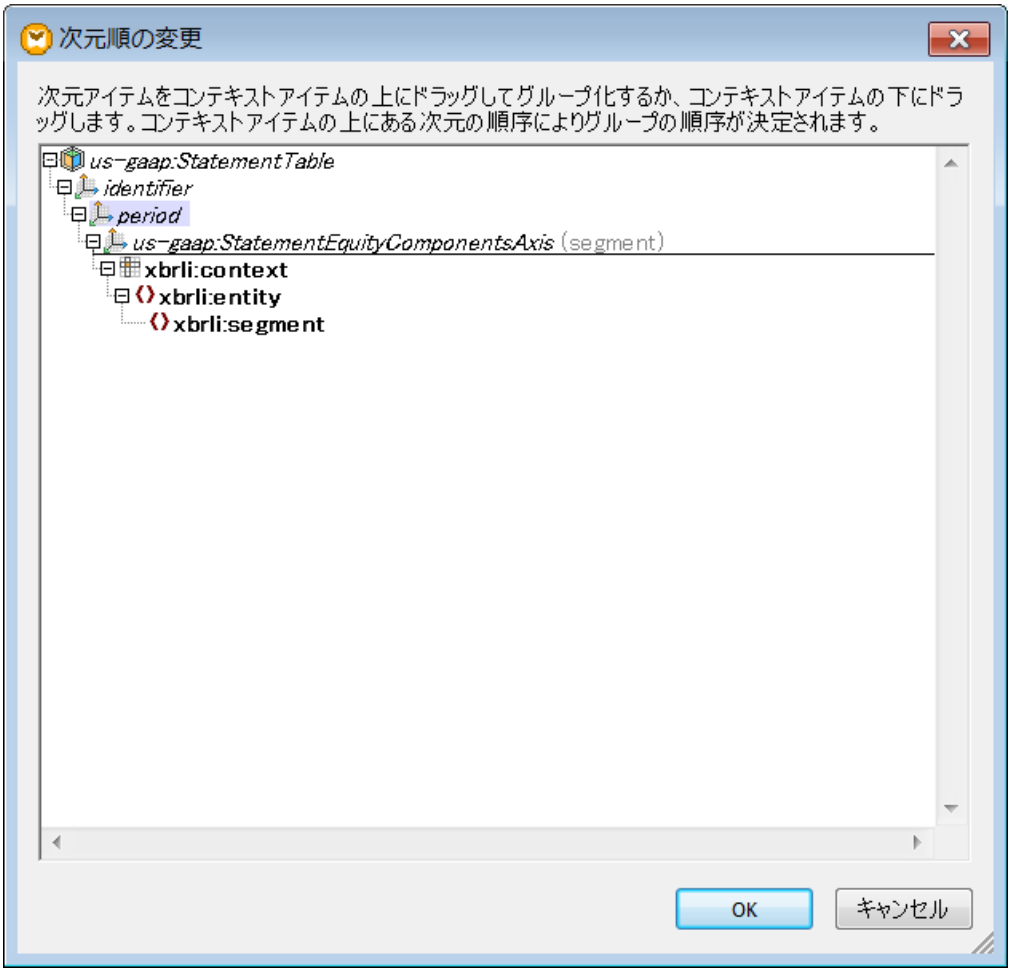
更に、特定のコンセプトに対して次元の値をセットする必要がある場合、MapForce では階層構造無しに次元を表示し、コンテキストノードの子要素として表示することができます。

次元アイテムの順序を変更する:

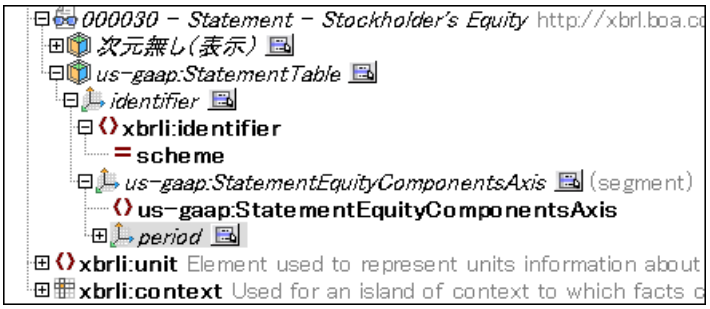
1. をクリックして、ポップアップメニューから次元順の変更エントリーを選択します。



この操作によりダイアログボックスが表示され、ハイパーキューブの次元位置を変更することができます。
ハイパーキューブには**identifier** と**period** という、2つの**デフォルト次元**があり、ハイパーキューブ内部におけるこれら次元の位置も変更することができます。
2. ハイパーキューブ次元をクリックして、ドラッグアンドドロップによりその次元を移動します。



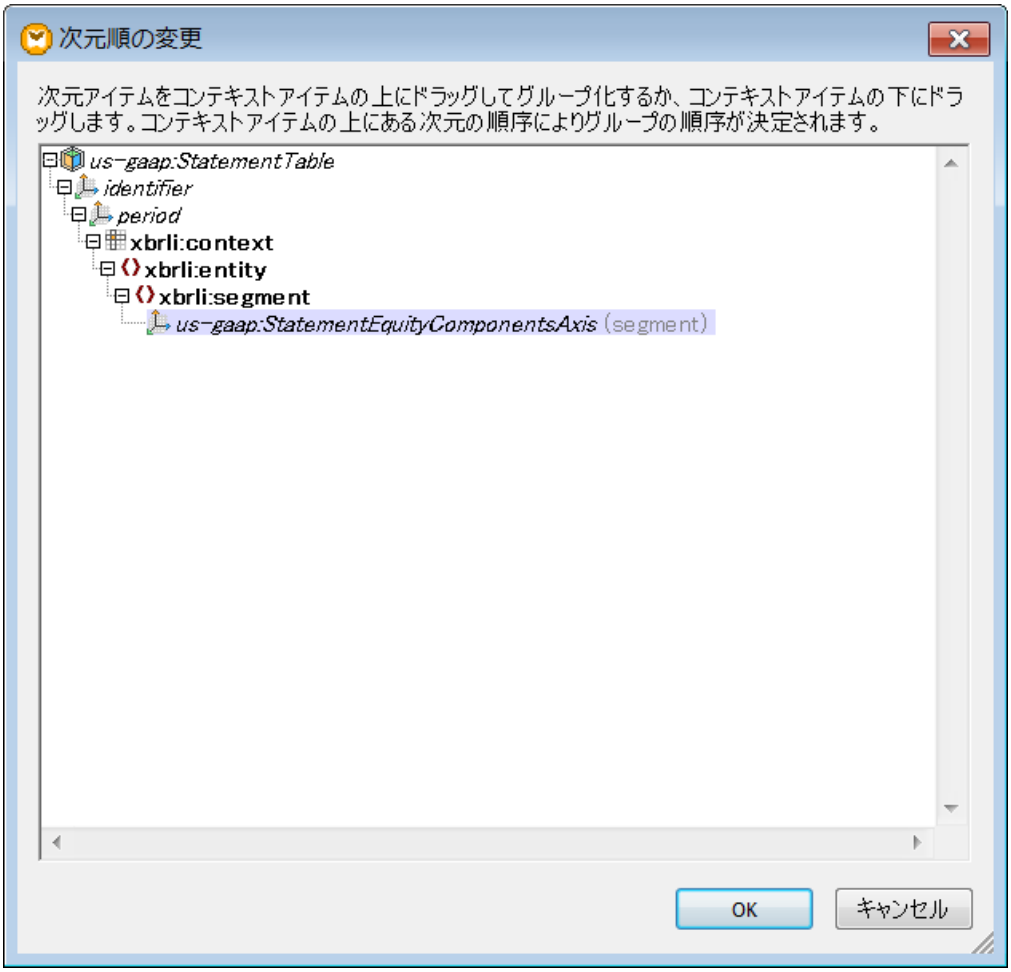
次元をドロップすることができる位置にラインが表示されます。



3. **「OK」**をクリックすると、ダイアログが閉じられ、コンポーネント内の次元位置が変更されます。

階層構造から次元を除外する

- 次元を**xbrli:context** ライン以下にドラッグすると、次元がコンテキストアイテムに挿入されます。



ハイパーキューブの明示的な次元のために Value-Maps を生成する

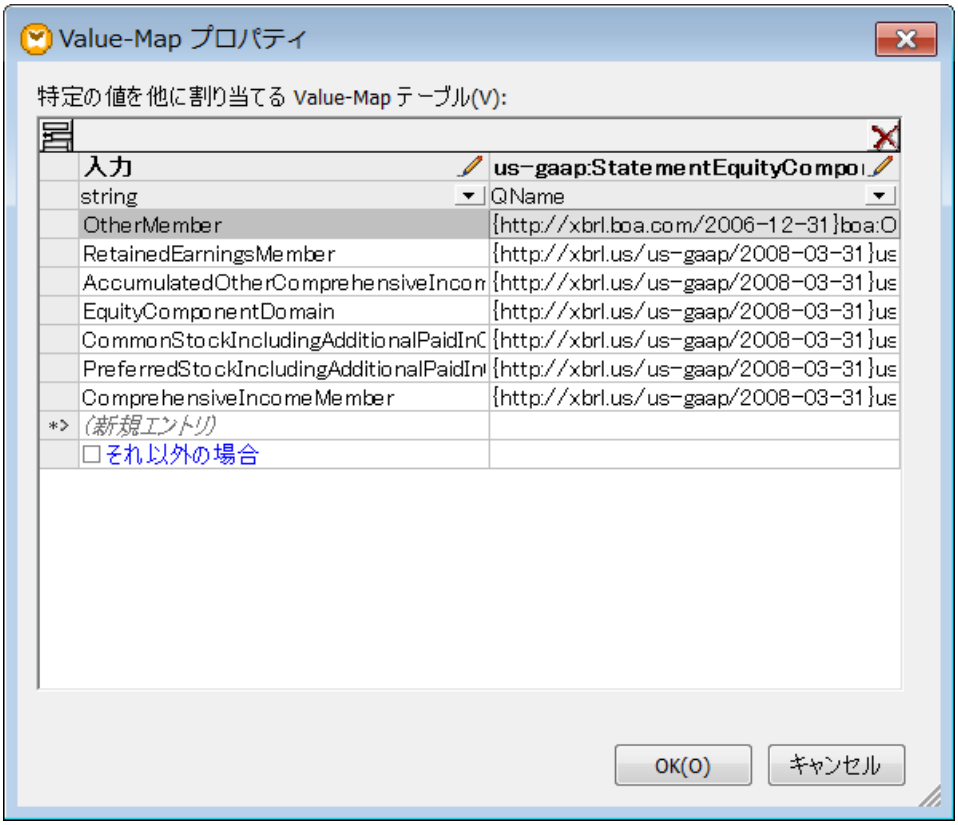
*このオプション*により、任意の型の入力データをターゲットコンポーネント内における妥当な QName へ変換します。つまり、入力文字列をプレフィックス付きの名前 (QName) へ変換します。

選択されたハイパーキューブ次元に対して Value-Map ルックアップテーブルを生成する

1. 次元値が特定のハイパーキューブ次元からは見えないようになっていることを確認してください。
2. 目的の**次元**にあるアイコン をクリックして、ポップアップメニューから入力として Value-Map を生成オプションを選択します。

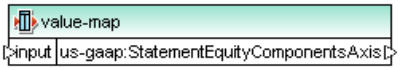


この操作により、Value-Map プロパティダイアログボックスが開かれ、タクソノミにて定義された通り、次元のデフォルトドメインならびにドメインメンバーをベースに、自動的に生成された入力ならびに出力値が表示されます。



3. 必要に応じて入力値を編集した後、「OK」をクリックしてValue-Map コンポーネントを挿入します。カラムのヘッダーテキストをヘッダーをダブルクリックした後に編集して、必要に応じて更に短い、または詳細な名前に変更することもできます。

4. この操作によりValue-Map 関数が挿入され、入力ならびに出力パラメーター名が表示されます。出力コネクタがターゲットのドメイン要素に対して自動的に接続されます。



5. 変換される入力データを含むソースアイテムを関数の入力パラメーターへ接続します。

一般的なValue-Map コンポーネントの使用方法についてはValue-Map - 入力データの変換を参照ください。

7.5.16.10 XBRL テーブルと作業する

XBRL タクソノミがテーブルリンクベース (Table Linkbase Specification 1.0 を参照してください)を参照する場合、MapForce は、このテーブルリンクベース(テーブルまたはブレークダウンなど)に適用することのできるノードの型を表示するように構成することができます。これにより、表示されるテーブルにデータをマップすることができます (MapForce によるテーブルリンクベースからの情報表示に関しての詳細は、構造ビューの選択を参照してください)。

Table Linkbase からの情報を表示するには、MapForce は通常、以下の例外を除いて、構造モデルを使用します :

- マージされたルールノードは表示されます
- 兄弟を持たないロールアップノードは、個別の階層レベルを作成しません。

構造モデルは、Table Linkbase Specification 1.0、§4 で定義されている3つのデータモデルうちの1つです。マッピングにより参照される構造モデルの一部を識別するために、MapForce は、定義モデルからの情報も使用します。

MapForce コンポーネント構造は一次元的ですが、テーブルは3つ(x、y、z)までの次元を持つことができ、MapForce は、テーブルの次元を全てのブレークダウンを共にネストすることにより、以下のデフォルトの順番 (z、y、x) で表示します。MapForce内のブレークダウンノードの目的は、各ブレークダウンの開始点に関する情報を提供するためです。

このセクションは、次のテーブルリンクベースからのエンティティとの作業方法の命令を含みます :

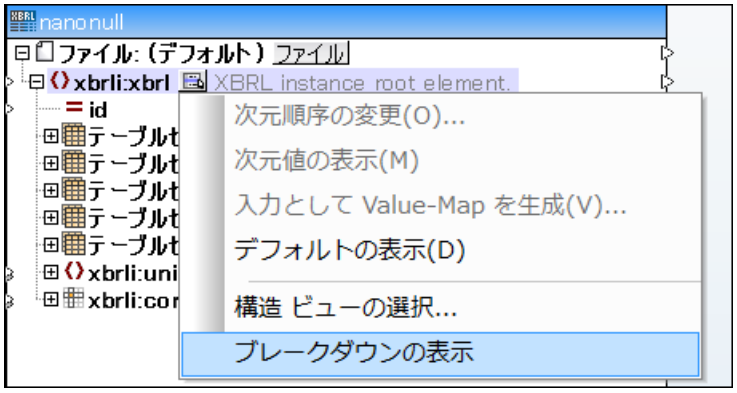
- ブレークダウンの表示または非表示
- ブレークダウンの順序の変更
- パラメーターと作業

ブレークポイントを表示または非表示

テーブルリンクベースを参照するXBRL ドキュメントでは、コンポーネント内でブレークダウンを表示、または、非表示するかを選択することができます。

ブレークポイントを表示または非表示

1. XBRL コンポーネント内で、**「コンテキストメニューの表示」**( ) ボタンが右横に表示されているノードを探します (例 :root node)。



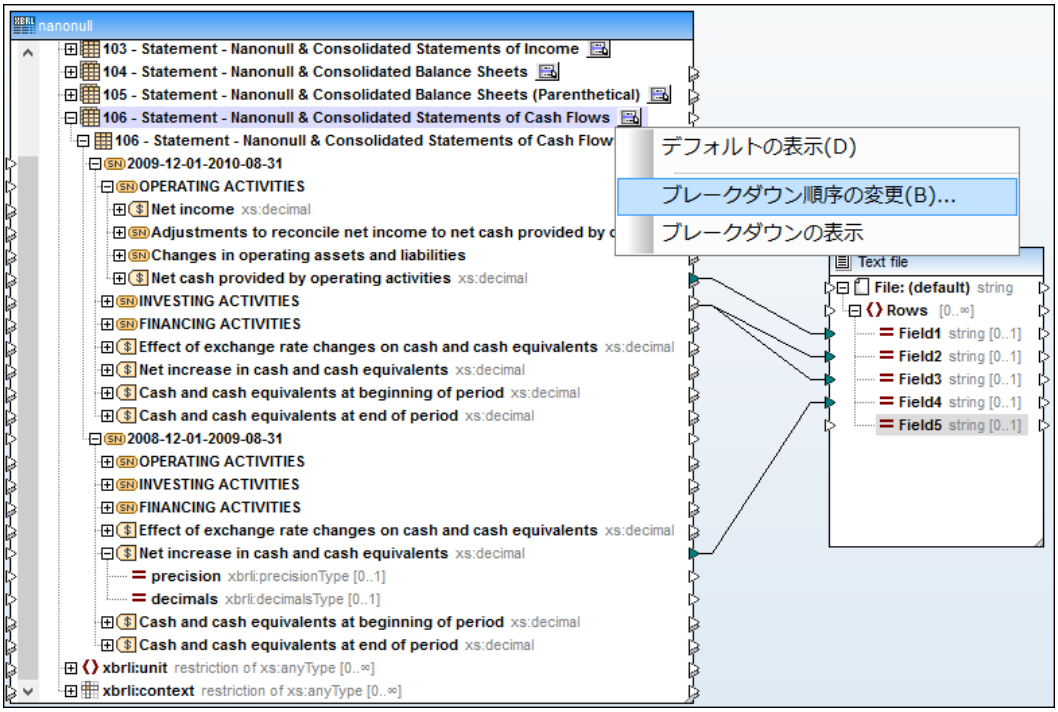
2. **「コンテキストメニューの表示」**( ) ボタンをクリックして、**「ブレークダウンの表示」**をクリックします。

ブレークダウンの順序の変更

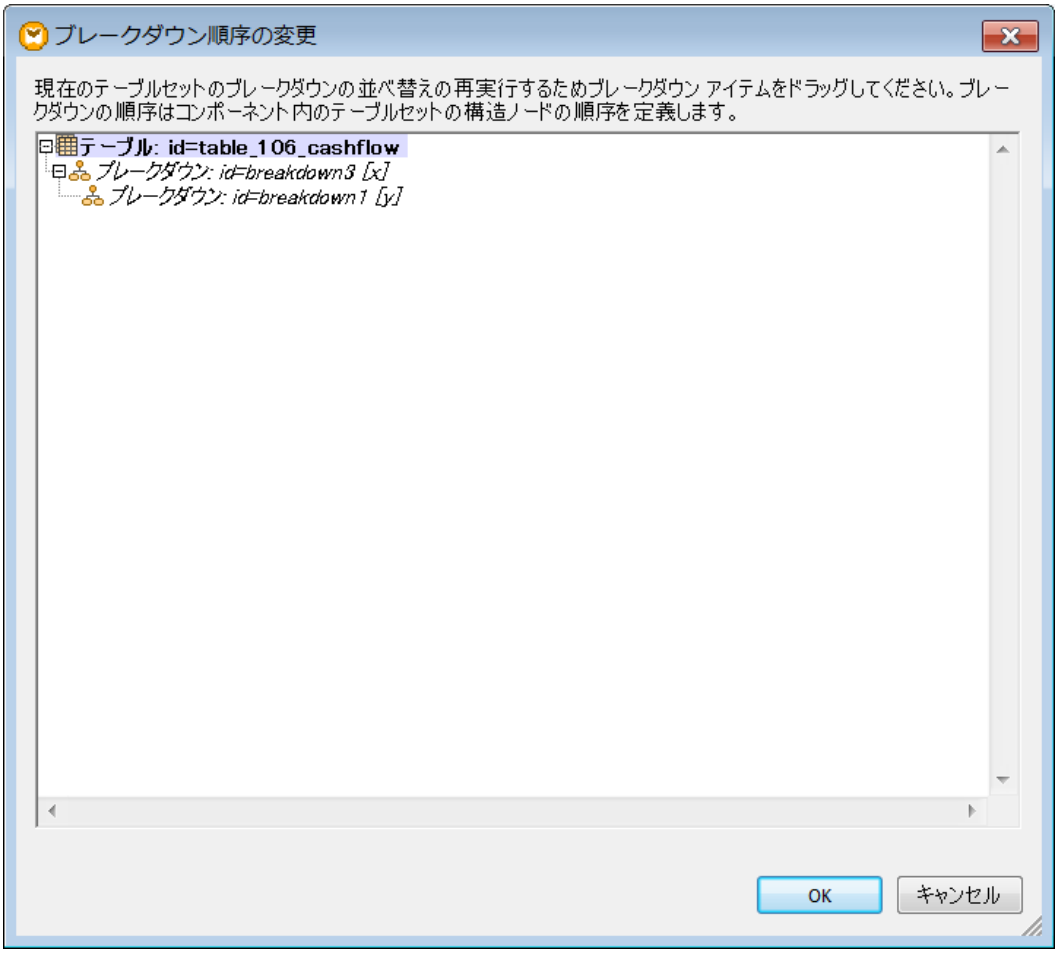
*テーブルリンクベース*を参照するXBRL ドキュメントでは *ブレークダウン*の順序を変更することができます。

ブレークダウンの順序の変更方法

1. XBRL コンポーネント内で*ブレークダウン*の順序を変更するノードをロケートします。
2. ノードの右にある **「コンテキストメニューの表示」**（ ）ボタンをクリックして、 **「ブレークダウンの順序の変更」** をクリックします。（ **「ブレークダウンの順序の変更」** メニューオプションが使用することができない場合、これは選択されたノードにとりこのオプションが有効でないことを意味します）。



3. *ブレークダウン*を希望する場所にドラッグします（例：以下のスクリーンショットでは、*y ブレークダウン*を*x ブレークダウン*にドラッグします）。



4. **「OK」**をクリックします。

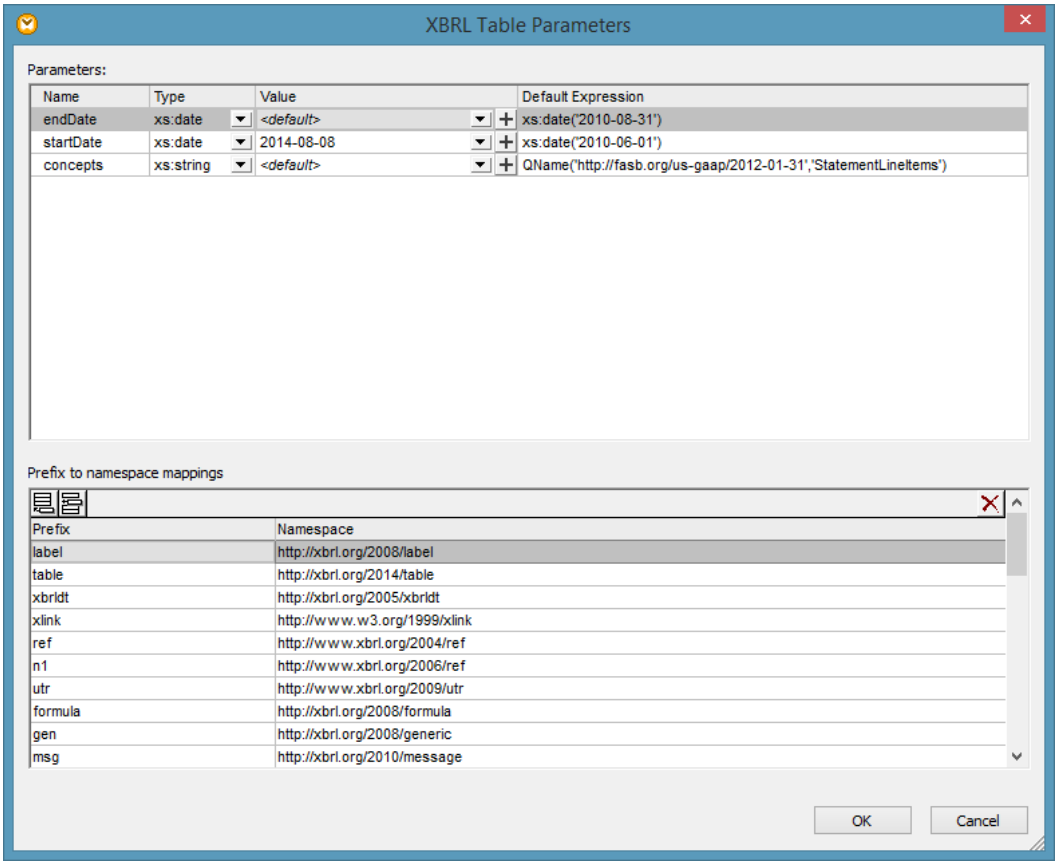
パラメーターと作業

XBRL ドキュメントがテーブルリンクベースを参照し、パラメーター (Table Linkbase Specification 1.0, §5.3) を含む場合、パラメーターの値を変更することができます。コンテキストにサポートされる場合、新しいパラメーター値を既存のパラメーターに追加することができます。

XBRL テーブルパラメーター ダイアログボックスからパラメーター値を変更することができます (下の命令を参照)。このダイアログボックスは、グローバルレベル (検知できるタクソノミセット内どこでも) と個別のテーブル内で定義されるパラメーターを表示します。パラメーターの値を変更し、ダイアログボックスを閉じると、MapFore は、コンポーネント内のすべてのテーブルを再生成し、構造ツリーノードをマージしようと試みます。パラメータが不足するために再生成が行われない場合、コンポーネントの対応するノード内にエラーノードが表示されます。エラーに関する詳しい情報は、 メッセージウィンドウのチェックボタンをチェックします。

パラメーターの値の変更

1. ルートノードの右の **「コンテキストメニューの表示」** ( ) ボタンをクリックして、 **「パラメータの定義」** をクリックします。(**「パラメータの定義」** メニュー オプションが使用できない場合、テーブルリンクベースはパラメーターを含みません)。



2. 新しい列内の新しい値のパラメーターレコードを選択します。

既存のパラメーターにパラメーター値を追加する

Select パラメーター レコードを選択して、値 列の下の **「値の追加」**（ ）ボタンをクリックします。

追加された値を削除する

削除する値の横の **「値の削除」**（ ）ボタンクリックします。

7.5.16.11 XBRL マッピングのサンプル

このセクションは、以下のXBRL マッピングのサンプルを含みます：

- Microsoft Access からXBRL へマッピング (Microsoft Access データベースからのデータがXBRL タクソノミにマップされる方法を示し、有効なXBRL インスタンスファイルを生成します。)
- Microsoft Excel からXBRL へマッピング (Microsoft Excel スプレッドシートからのデータがXBRL タクソノミにマップされる方法を示し、有効なXBRL インスタンスファイルを生成します。)

Microsoft Access からXBRL へマッピング

データベースのデータを抽出するための様々な関数やフィルターを使用したこのサンプルは...\MapForceExamples フォルダー内に収められている**DB_to_XBRL.mfd** ファイルにて確認することができます。

タクソノミの**nanonull.xsd** はUS:GAAP から派生したものです。マッピングにより、1つのDisclosure と3つのStatement に対して、全てのコンテキスト、コンセプト、ユニット、そしてフットノートを含むXBRL 出力インスタンスが作成されます。

レポート**"091 - Disclosure - Segment Revenue and Operating Income"** では、MapForce で次元の値をマッピングする方法が示されます。ハイパーキューブの "us-gaap:ScheduleOfSegmentReportingInformationBySegmentTable" には明示的な次元の "us-gaap:StatementOperatingActivitiesSegmentAxis" が含まれています。

このドメインは、"nanonull:USA"、"nanonull:Europe"、そして "nanonull:Asia" という3つの次元値によりタクソノミ内にて拡張されます。マッピングでは、Value-Map によりデータベーステーブルの "Region" にある "Name" カラムの値が、QName 型で目的の次元値へ変換されます。

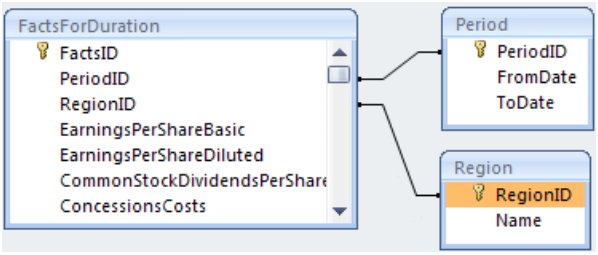
レポート**"106 - Statement - Nanonull and Consolidated Statement of Cash Flows"** では、MapForce で期間とインスタンス期間の両方に関連した出力インスタンス内へ、ファクトを書き込む方法が示されます。

マッピングに示される通り、階層構造におけるperiod アイテムを複製することで "Cash and cash equivalents at beginning (end) of period" といったファクトが作成されます。

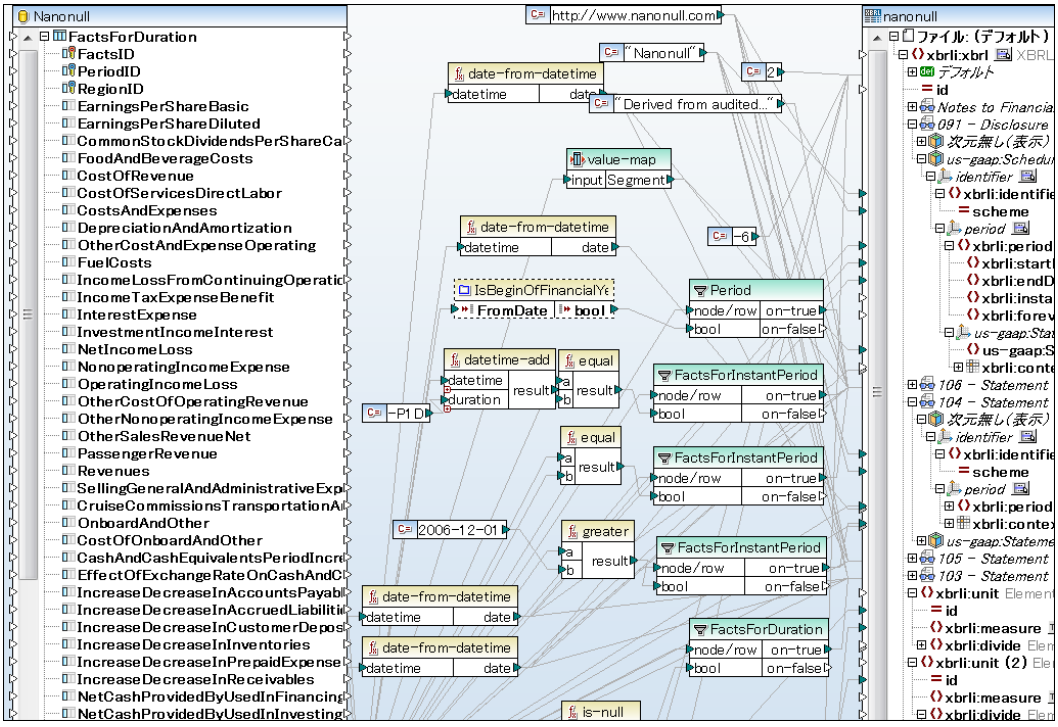
マッピングにより出力インスタンス内にて、"USD" と "perShareItemType" という2つのユニットが定義されます。このためには、xbrli:unit 要素を複製する必要があります。XBRL ライブラリから "xbrl-measure-currency" と "xbrl-measure-shares" 関数を使用することで、関連するmeasure 要素が作成されます。

instant または duration 期間に関連するかにより、データベーステーブル内のファクトが分割されます。

- **FactsForInstantPeriod** テーブルは、フラットな値のテーブルになります。
- **FactsForDuration** テーブルは階層構造になっていて、中に含まれる各ファクトが、特定のPeriodID ならびにRegionID に関連付けられます。



Period テーブルではFromDate ならびにToDate により、期間の開始ならびに終了日が定義されます。Region テーブルにより、ファクトが各地域（Asia、Europe、またはUSA）へ関連付けられます。



XBRL *インスタンスファイル*では、以下のXBRL *アイテム*が必要になります：

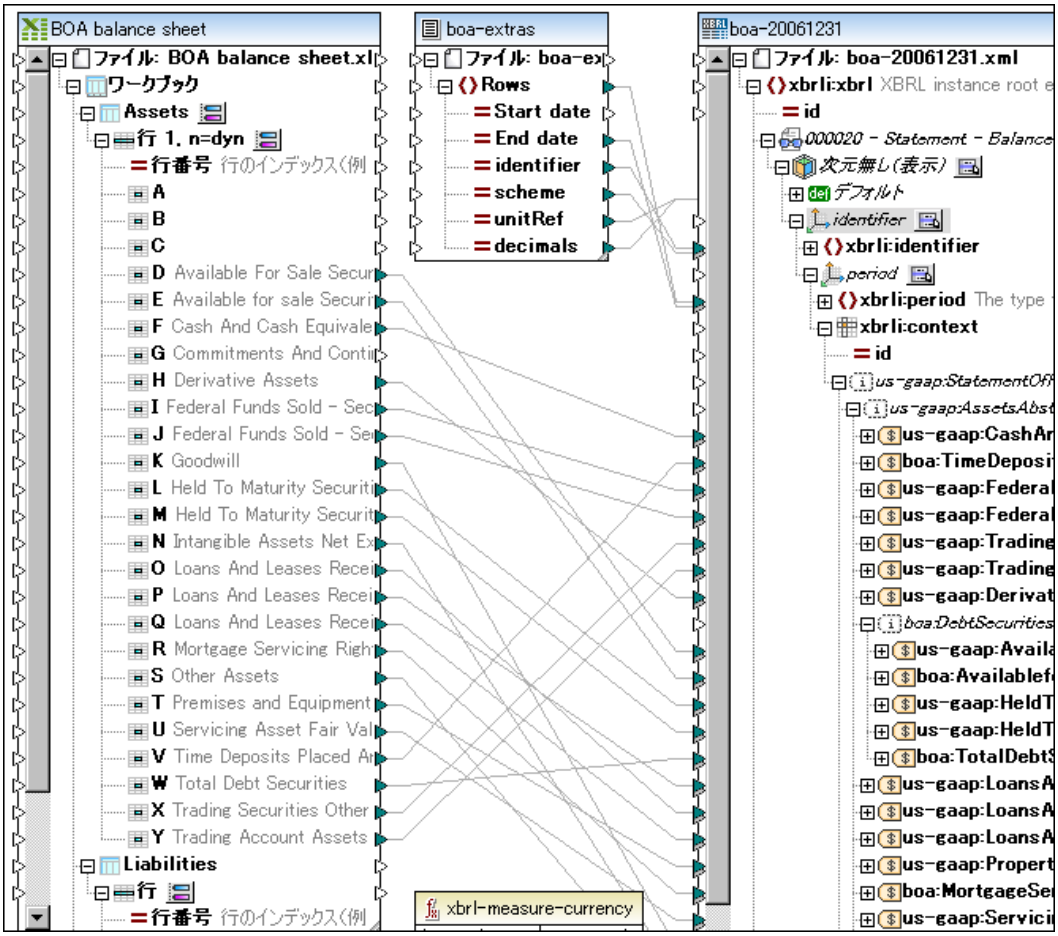
- **unitRef** と monetary コンセプト内の**decimals** または**precision** のどちらか
- *次元の***スキーム**とxbrli:**identifier**
- xbrli:**period** と xbrli:instant または xbrli:**startDate**/xbrli:**endDate** 要素のどちらか
- xbrli:**unit** 要素内における xbrli:**measure** ならびに xbrli:**id**

Microsoft Excel からXBRL へマッピング を参照してください。

Excel からXBRL へのサンプル

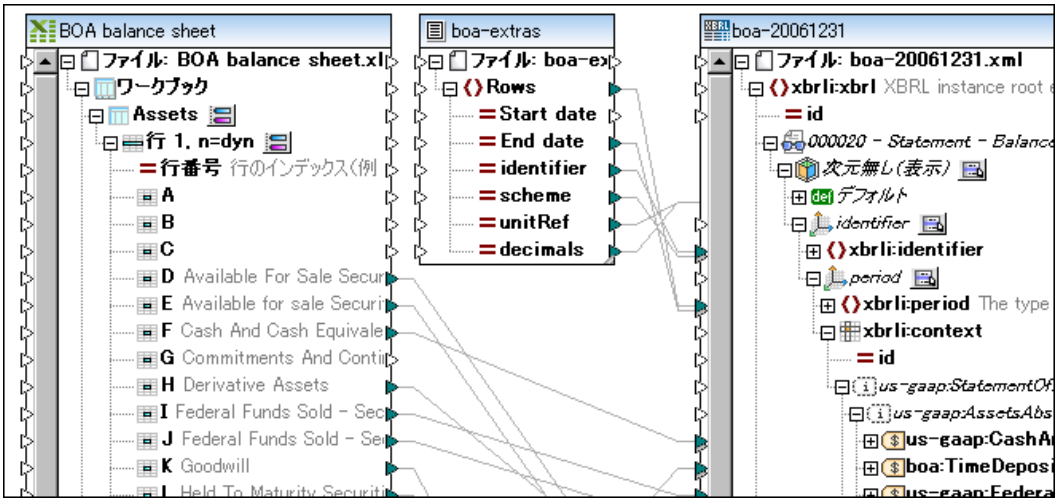
*このセクション*で使用される*マッピングファイル*は、...\MapForceExamples\Tutorial\ フォルダーに収められている**boa-balance-sheet.mfd** から確認することができます。*このサンプル*で使用される*タクソノミファイル*は .\Tutorial フォルダーに収められています。

以下に示されるサンプルでは、Excel シートからXBRL *タクソノミ*へのマッピングを行い、XBRL *インスタンスファイル*を生成する方法を確認することができます。マッピングをシンプルに保つため、"Assets" ワークシートから得られたデータだけがマッピングされています。Dec. 31st 2006 という*インスタンス*に対してAssets を含んでいる、妥当なXBRL *インスタンス*がマッピングの結果として生成されます。



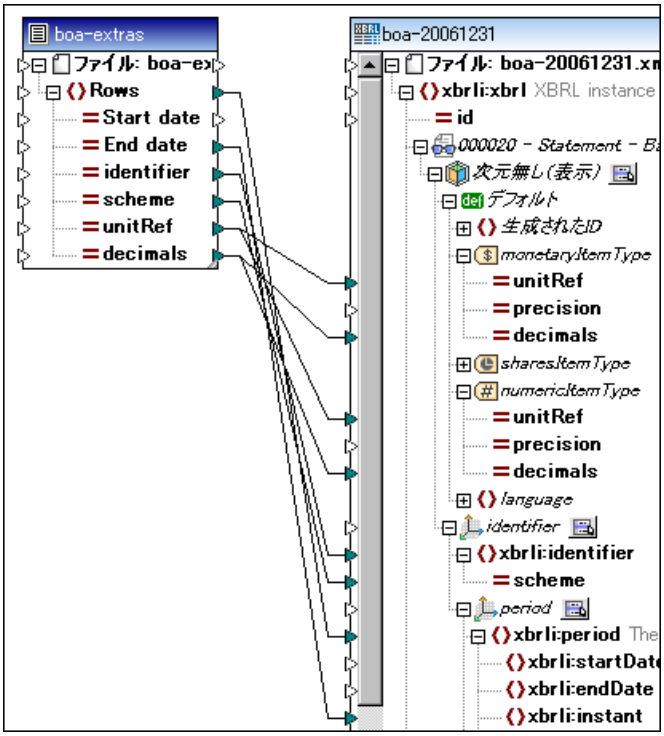
このサンプルでは、アルファベット順で表示されているExcel の列名を、タクソノミコンポーネント内にあるXBRL アイテム/ファクト (Cast ならびにCash Equivalents) へ直接マッピングする方法について説明します。

ソースコンポーネントとターゲットコンポーネント内にあるアイテムの名前を一致させる必要は無いという点に留意してください。ターゲットタクソノミアイテムの名前により、結果 XBRL インスタンス内で使用される名前が決定されます。ソースとターゲットアイテムの名前が一致している場合、一致した名前の子アイテムを自動的に接続する、**子の自動接続機能を**利用することができます (マッチする子要素の接続を参照)



妥当なXBRL インスタンスファイルを生成するためには、多くのマッピングに対してマッピングを行う必要があります。一部の必須アイテムはテキストファイル (boa-extras) により与えられており、テキストファイルのフィールドと、次元 (表示) ハイパーキューブ内に子アイテムがある "Default" アイテムがマッピングされます。

ハイパーキューブ次元を右クリックして、「XBRL | デフォルトユニットの表示」を選択するか、アイコン をクリックして表示される同名のオプションを選択することで、"Default" アイテムを挿入することができます。



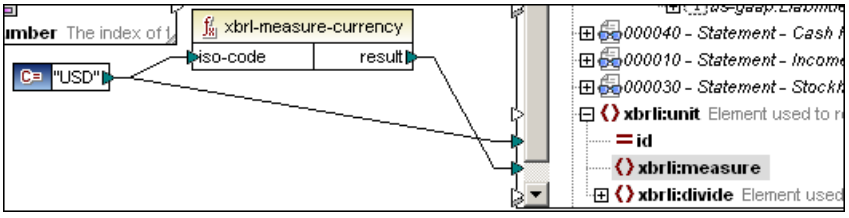
XBRL インスタンスファイルにて必要なXBRL アイテムを以下に示します：

- **unitRef** と monetary コンセプト内にある**decimals** または**precision**
- identifier 次元の**スキーマ**ならびにxbrli:**identifier**
- xbrli:**period** と xbrli:instant または xbrli:**startDate**/xbrli:**endDate** 要素のどちらか
- xbrli:**unit** 要素内にあるxbrli:**id** ならびにxbrli:**measure**

boa-extras テキストファイルからは、以下に示されるその他必須アイテムのデータが与えられます。



定数コンポーネントにより、**id** ならびに**measure** アイテムのデータが与えられ、タクソノミコンポーネントのベースに使用されます。



出力ボタンをクリックして、XBRL インスタンスファイルを確認します。

- ツールバーにある出力の検証ボタン をクリックすることで、XBRL インスタンスの検証を行うことができます。メッセージや警告がメッセージウィンドウにて表示されます。
- テキストビュー設定ボタン をクリックすることで、出力ビューの設定を行うことができます。

```
<?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.o
  <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</iden
    </entity>
    <period>
      <instant>2006-12-31</instant>
    </period>
  </context>
  <us-gaap:CashAndCashEquivalentsAtCarryingValue contextRef="ctx1" unitRef="USD" decimals="-6">36429</us-gaap:CashAn
  <boa:TimeDepositsPlacedAndOtherShorttermInvestments contextRef="ctx1" unitRef="USD" decimals="-6">13952</boa:TimeDep
  <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResell contextRef="ctx1" unitRef="USD" decimals="-6
  <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResellPledgedAsCollateral contextRef="ctx1" unitRef=
  <us-gaap:TradingSecuritiesOther contextRef="ctx1" unitRef="USD" decimals="-6">153052</us-gaap:TradingSecuritiesOther>
  <us-gaap:TradingSecuritiesPledgedAsCollateral contextRef="ctx1" unitRef="USD" decimals="-6">922274</us-gaap:TradingSecu
  <us-gaap:DerivativeAssets contextRef="ctx1" unitRef="USD" decimals="-6">23439</us-gaap:DerivativeAssets>
  <us-gaap:AvailableForSaleSecuritiesDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">192806</us-gaap:Availabl
  <boa:AvailableforsaleSecuritiesDebtSecuritiesCollateral contextRef="ctx1" unitRef="USD" decimals="-6">83875</boa:Availablef
  <us-gaap:HeldToMaturitySecurities contextRef="ctx1" unitRef="USD" decimals="-6">40</us-gaap:HeldToMaturitySecurities>
  <us-gaap:HeldToMaturitySecuritiesFairValue contextRef="ctx1" unitRef="USD" decimals="-6">40</us-gaap:HeldToMaturitySecu
  <boa:TotalDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">192846</boa:TotalDebtSecurities>
  <us-gaap:LoansAndLeasesReceivableGrossCarryingAmount contextRef="ctx1" unitRef="USD" decimals="-6">706490</us-gaa
  <us-gaap:LoansAndLeasesReceivableAllowance contextRef="ctx1" unitRef="USD" decimals="-6">9016</us-gaap:LoansAndLe
  <us-gaap:LoansAndLeasesReceivableNetReportedAmount contextRef="ctx1" unitRef="USD" decimals="-6">697474</us-gaap:L
  <us-gaap:PropertyPlantAndEquipmentNet contextRef="ctx1" unitRef="USD" decimals="-6">9255</us-gaap:PropertyPlantAndEqu
  <boa:MortgageServicingRights contextRef="ctx1" unitRef="USD" decimals="-6">3045</boa:MortgageServicingRights>
  <us-gaap:ServicingAssetAtFairValueAmount contextRef="ctx1" unitRef="USD" decimals="-6">20</us-gaap:ServicingAssetAtFa
  <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:IntangibleAss
  <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>
```

7.5.17 HL7 Version 3

HL7 バージョン 3x は XML をベースとしたものであるため、MapForce 2017 にて自動的にサポートされます

HL7 V2.2 - V2.5.1 XML スキーマならびに構成ファイルに対応したインストーラーは、Altova Web サイトの (https://www.altova.com/ja/components_mapforce.html) MapForce ライブラリページからダウンロードすることができます。HL7 V3 コンポーネントや XML スキーマをインストールするには、カスタムセットアップを選択してください。

インストール後の HL7 XML スキーマは以下の場所に配置されます：

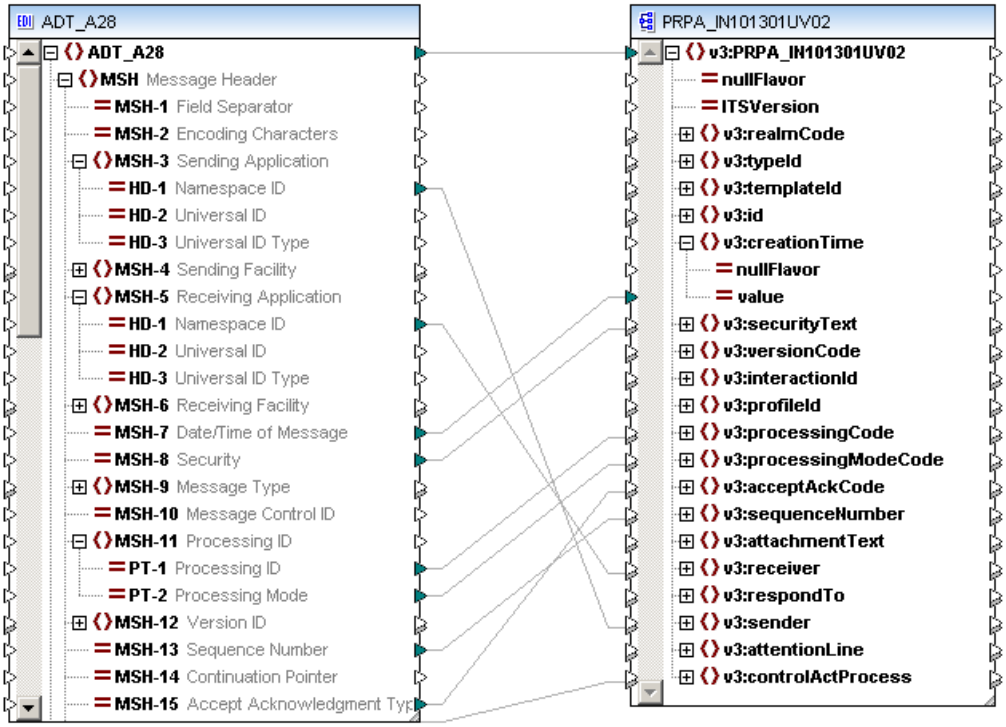
32-ビット オペレーティングシステム上の32-	C:\Program Files\Altova\Common2017\Schemas\hl7v3

ビット MapForce または 64-ビット オペレーティングシステム上の64-ビット MapForce	
64-ビット オペレーティングシステム上の32-ビット MapForce	C:\Program Files(x86)\Altova\Common2017\Schemas\hl7v3

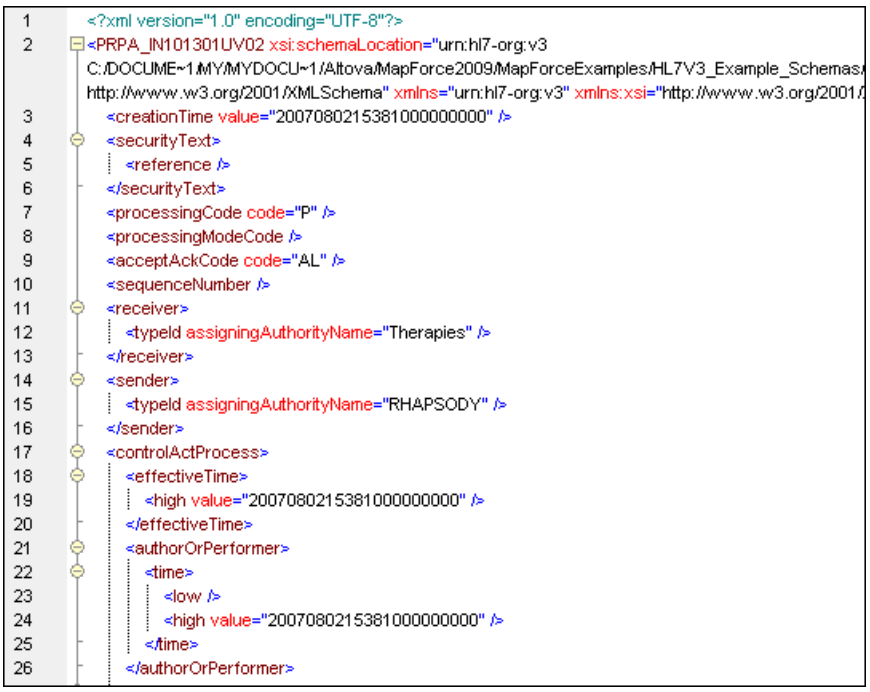
MapForce において、HL7 ドキュメントはソースならびにターゲットコンポーネントとして使用することができます。このデータと XML スキーマ、データベース、EDI またはその他の コンポーネントをマッピングすることができます。

このサンプルでは、 .\MapForceExamples フォルダー内に収められているHL7V260_To_HL7V3.mfd を使用します。HL7 V2.6 ドキュメントから、XML をベースにしたHL7 V3 へのマッピングが行われます。

- ...**MapForceExamples** フォルダーにはADT_A28 EDI ファイルが収められています。
- ターゲットXSD ファイルのPRPA_IN101301UV02.xsd は ..**MapForceExamples\HL7V3_Example_Schemas** フォルダーに収められています。



以下にマッピングの結果として生成されるXML インスタンスファイルを示します：



```
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="20070802153810000000000" />
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="20070802153810000000000" />
20      </effectiveTime>
21      <authorOrPerformer>
22        <time>
23          <low />
24          <high value="20070802153810000000000" />
25        </time>
26      </authorOrPerformer>
```


(C) 2017 Altova GmbH

Chapter 8

関数

8 関数

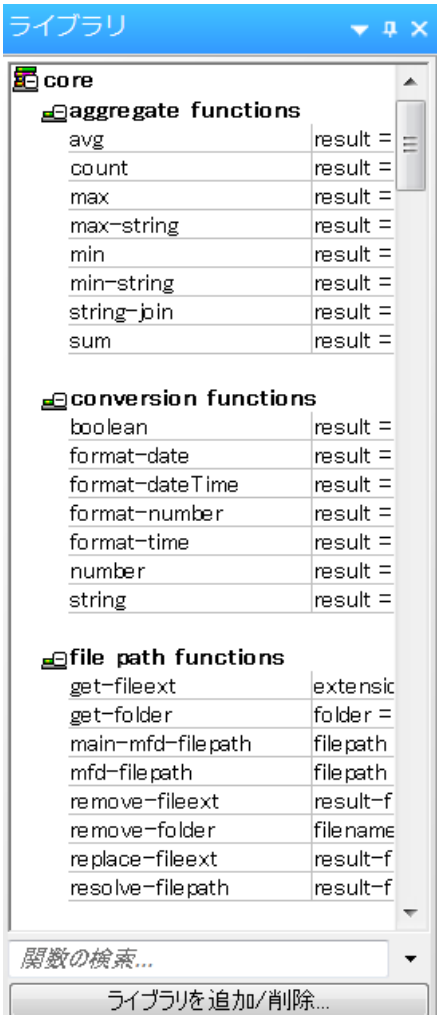
関数は、特定のノードに従いデータを変換するパワフルな方法を示します。 このセクションは、(MapForce のビルドインの関数または、外部のソースからの再利用にかかわらず) 関数との作業方法について説明します。以下のロードマップを使用して特定のタスクに関連した関数にアクセスしてください:

...を実行するには	以下を参照してください...
MapForce 内の関数との作業方法を学ぶ	関数と作業
マッピング内のオペレーションの優先順位の変更方法を学ぶ	優先コンテキスト ノード アイテム
MapForce 内で自身の関数を作成する	ユーザー定義関数
外部 XSLT 関数を使用する	カスタム XSLT 関数の追加
外部 .NET .dll とJava クラスライブラリを使用する	カスタム Java、C# とC++ 関数ライブラリの追加
MapForce で使用するために自身のJava ライブラリに書き込む	Java ライブラリの作成
MapForce で使用するために自身のC# ライブラリに書き込む	C# ライブラリの作成
MapForce で使用するために自身のC++ ライブラリに書き込む	C++ ライブラリの作成
すべてのビルドインMapForce 関数を表示する、または特定の関数の詳細を検索する	関数ライブラリ リファレンス

8.1 関数と作業

マッピング内で関数を使用する：

1. 変換言語を選択する（変換言語の選択を参照してください）。
2. ライブラリウィンドウ内の必要な関数をクリックし、マッピングエリアにドラッグします。

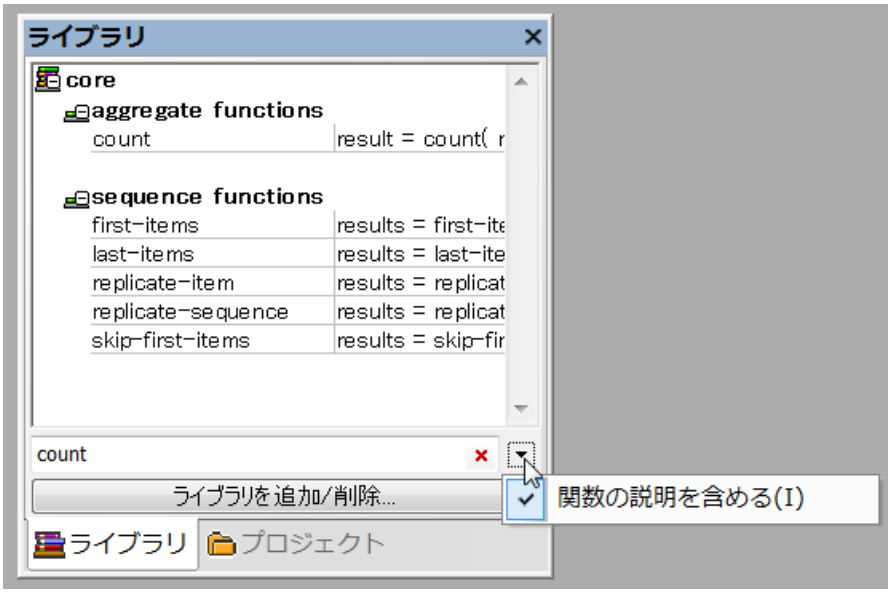


ライブラリウィンドウ内で関数を検索する：

1. ライブラリウィンドウの下の部分にある、テキストボックスに関数名を入力します。



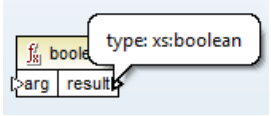
デフォルトでは、MapForceは、関数を名前と説明テキストにより検索します。関数の説明を検索から除外する場合は、**関数説明を含める**オプションを無効化します。



検索をキャンセルする場合、「**Esc**」キーを押す、または × をクリックします。

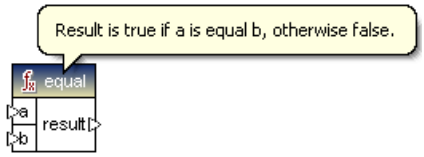
関数入力 または 出力引数のデータ型を確認する:

1. 「**ヒントの表示**」 ツールバーボタンが有効化されていることを確認してください。
2. 関数の引数部分にマウスを移動します。



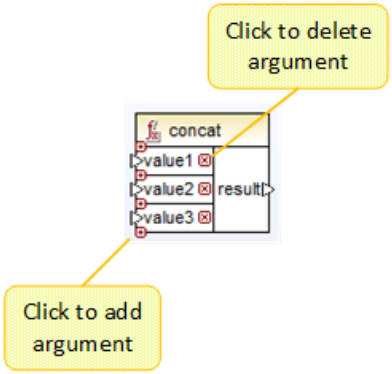
関数の詳細を表示する：

1. **「ヒントの表示」** ツールバーボタンが有効化されていることを確認してください。
2. 関数の上にあるマウスのポイントを移動します（『ライブラリ』ペインとマッピングエリアの両方でこの機能を使用することができます）



(適用することのできる関数のための) 関数引数の追加と削除 ：

- 追加または削除するパラメーターの横の **「パラメーターの追加」**（⊙）または **「パラメーターの削除」**（⊗）をクリックします。



⊙ シンボルに接続をドロップすると、パラメーターが自動的に追加され、接続します。

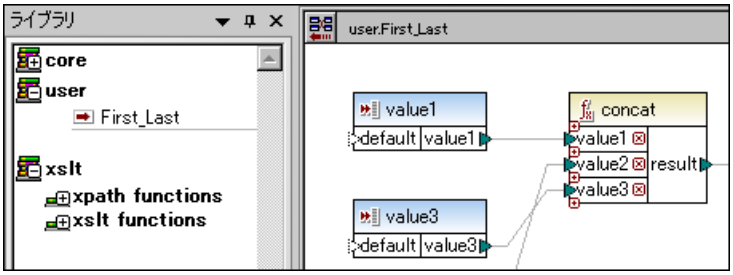
現在アクティブなマッピング内で関数の全ての発生を検索する：

- ライブラリウィンドウを右クリックして、コンテキストメニューから **「全ての呼び出しの検索」** を選択します。検索の結果はメッセージウィンドウに表示されます。

8.2 ユーザー定義関数

MapForce ではメインマッピングウィンドウと同じように、視覚的な方法によりユーザー定義関数を作成することができます。

これらの関数はライブラリウィンドウにて利用できるようになり（例：First_Last）、デフォルトで用意されている関数と同様の方法で使用することができます。関数を使用することで、ブロック単位でマッピングの管理を行い、別のマッピングにて再利用することができます。



ユーザー定義関数は、メインのマッピングとともに ***.mfd** ファイルに収められます。

ユーザー定義関数では、**入力**ならびに**出力**コンポーネントを使用することで、メインマッピング（又は他のユーザー定義関数）からユーザー定義関数へ情報が受け渡され、ユーザー定義関数におけるマッピングの結果が返されます。

ユーザー定義関数では、例えばルックアップ関数を実装する際に便利な XML スキーマやデータベースのような、（ユーザー定義関数内部で使用される）「ローカル」ソースコンポーネントを含めることができます。

ユーザー定義関数には任意の数の入力ならびに出力を含めることができ、これら入出力コンポーネントは、単純型の値、XML ノード、データベース、EDI ファイル、または FlexText 構造ファイルとして定義することができます。

ユーザー定義関数は、以下のような状況で利用することができます：

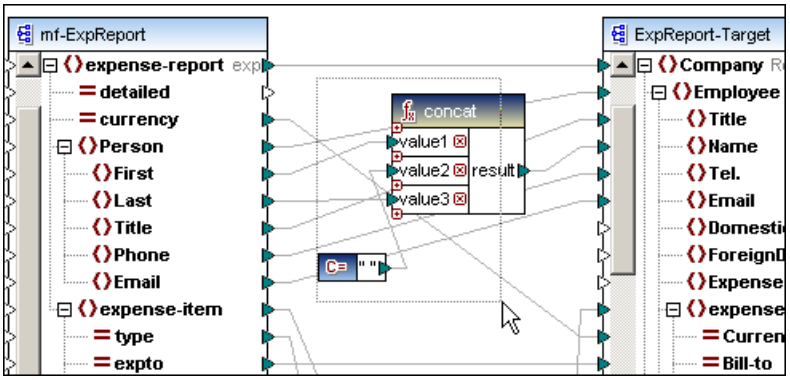
- 複数の関数を 1 つのコンポーネントへまとめる。例えば、特定のフィールドをフォーマットしたり、値のルックアップを行う
- これらコンポーネントを任意の数だけ再利用する。
- （マッピングファイルをライブラリとしてロードすることで）他のマッピングからユーザー定義関数をインポートする。
- インライン関数 を使用することで、複雑なマッピングを個々に編集可能な細かいパーツへ分割する。
- 再帰的なユーザー定義マッピング を作成することで、再帰的なスキーマをマッピングする。

ユーザー定義関数は何も無い状態から作成することができるほか、マッピングタブに既に表示されている関数からも作成することができます。

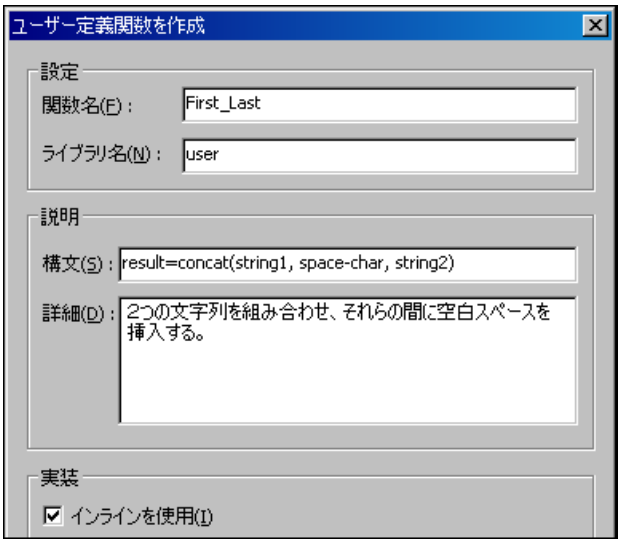
以下のサンプルは …\MapForceExamples\Tutorial\ フォルダーに収められている**Tut-ExpReport.mfd** ファイルを使用します。

既存のコンポーネントからユーザー定義関数を作成する：

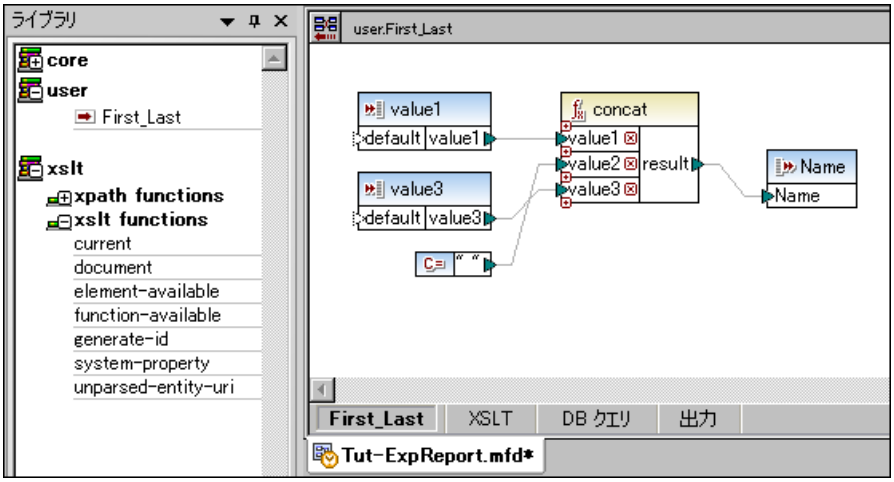
1. マウスポインターをドラッグして、"concat" と定数コンポーネントを選択します（Ctrl キーを押下しながらコンポーネントを複数選択することもできます）。



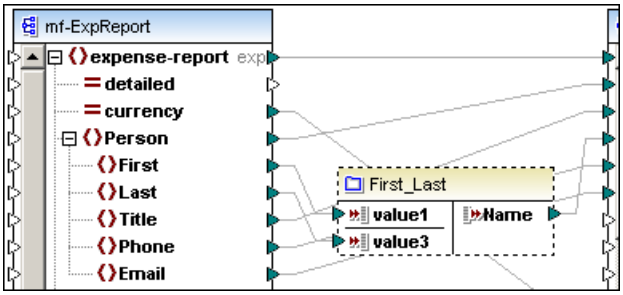
2. メニューオプションから「関数 | 選択からユーザー定義関数作成」を選択します。
3. 新たなユーザー定義関数の名前を入力します（例：First_Last）。
   メモ：ユーザー定義関数の名前に使用可能な文字は、大文字と小文字のアルファベット（a-z, A-Z）、数値（0-9）、そしてアンダースコア（_）、ハイフォンダッシュ "-"と コロン ":" となります。
4. 構文ならびに詳細フィールドにて、新たな関数に関する情報を入力することができ「OK」ボタンをクリックすることで確定します。入力されたテキストは、関数上にマウスオーバーされた際にツールチップとして表示されます。"user" というライブラリ名がデフォルトで与えられます。勿論、このフィールドで独自のライブラリ名を定義することもできます。



関数グループを構成する個々の要素が、関数名のタブに表示されます。新たに作成されたライブラリ "user" がライブラリペインに表示され、その下に "First_Last" という関数名が表示されます。



ホームボタン をクリックすることで、メインマッピングウィンドウへ戻ることができます。複数あったコンポーネントが First_Last という名前で 1 つの関数コンポーネントへ統合されたのか確認できます。入力ならびに出力パラメーターは自動的に接続されていることが確認できます。



インラインのユーザー定義関数は破線の境界線により表示されます。詳細に関しては[インラインユーザー定義関数](#) を参照ください。

ライブラリペインからマッピングウィンドウへ関数名をドラッグアンドドロップすることで、その関数をマッピングにて使用することができます。別のマッピングで関数を使用する方法については、[ユーザー定義関数の再利用](#) を参照ください。

ユーザー定義関数を開く

ユーザー定義関数を開くには以下を行います：

- ユーザー定義関数コンポーネントのタイトルバーをダブルクリックする。
- ライブラリペインにて、目的のユーザー定義関数をダブルクリックする。

これにより、関数の中にあるコンポーネントが、関数名のタブにて表示されます。ホームボタン をクリックすることで、メインのマッピングへ戻ることができます。異なる *.mfd ファイルのユーザー定義関数を (メインマッピングウィンドウにて) ダブルクリックすることで、その MFD ファイルが新たなタブにて表示されます。

ユーザー定義関数のナビゲーションを行う

様々なタブ (又はユーザー定義関数のタブ) を MapForce にてナビゲーションすることで、移動の履歴が自動的に生成され、進む/戻るアイコンをクリックすることで複数のタブ間を移動することができます。履歴はセッションごとに管理されており、複数の MFD ファイル間を移動することもできます。

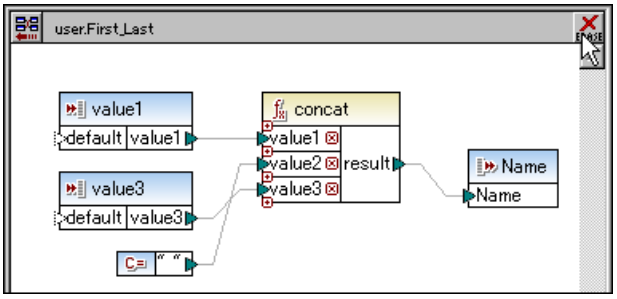
 ホームボタンをクリックすることで、ユーザー定義関数からメインマッピングのタブへ戻ります。

 戻るボタンにより、履歴内にある 1 つ前のマッピングへ移動します。

 進むボタンにより、履歴内にある 1 つ後のマッピングへ移動します。

ライブラリからユーザー定義関数を削除する：

1. ライブラリウィンドウにて目的のユーザー定義関数をダブルクリックします。
2. タイトルバーの右端にある **「削除」**ボタンをクリックすることで、その関数を削除することができます。



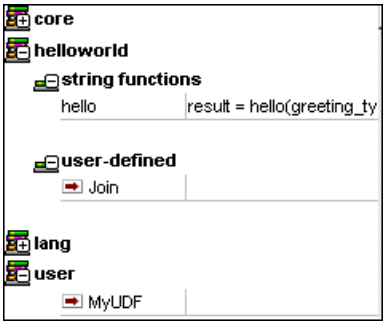
ユーザー定義関数を再使用 (インポート) する。

あるマッピングで定義されたユーザー定義関数は、他のマッピングにてインポートすることができます：

1. ライブラリウィンドウの下部に配置されている**ライブラリの追加/削除**をクリックします。
2. 追加ボタンをクリックし、インポートするユーザー定義関数を含んだ ***mfd** ファイルを選択します。ユーザー定義代わりに、関数を定義する際に、自由に名前を「ライブラリ名」にて入力することができます。(カスタム、Java、C# と C++ 関数ライブラリの追加を参照)。ユーザー定義関数はライブラリウィンドウに表示されます。デフォルトのライブラリ名を使用してユーザー定義関数が作成された場合、ライブラリ名は "user" です。それ以外の場合、ユーザー定義関数を作成した再に指定したライブラリ名を検索してください。
3. インポートされた関数をマッピングへドラッグすることで、その関数を利用することができます。

> ユーザーを作成する場合、複数の *mfd ファイル内で同じライブラリ名 または カスタムライブラリ (Adding カスタムライブラリを参照) を指定することができます。この場合、使用することのできるソースからの関数は、ライブラリウィンドウ内の同じライブラリ名の下にリストされます。しかしながら、現在アクティブな ドキュメント内の関数のみがダブルクリックにより編集することができます。

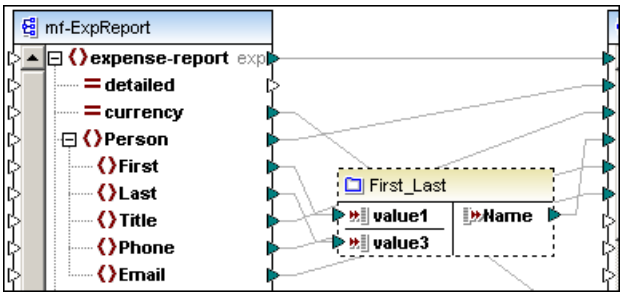
以下のサンプルを考慮してください：



- "helloworld" ライブラリ内の関数 "hello" はカスタムライブラリからインポートされます。
- 現在の*.mfd ファイルにて定義された "Join" という名前の関数が "helloworld" ライブラリ以下に表示されています。
- 現在の *mfd ファイルにて定義された "MyUDF" という名前の関数が "user" ライブラリ以下に表示されています。

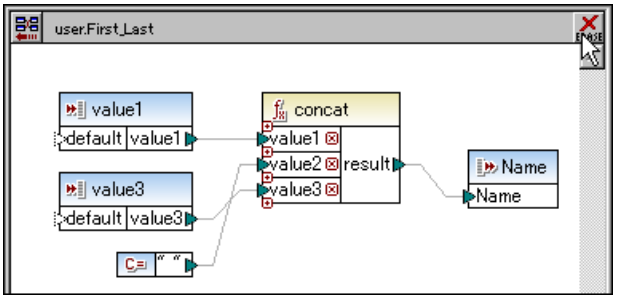
インポートされた関数内の変更はインポートされるマッピングにもライブラリ***.mfd** ファイルを保存する際に適用されます。

ユーザー定義関数内におけるパラメーターの順序



ユーザー定義関数内のパラメーターの順序は直接影響することができます：

- 入力ならびに出力パラメーターの順序は、上部から下部にかけて引数の位置（パラメーターコンポーネントの左上ピクセルの位置）により決定されます。
- 2 つのパラメーターが垂直の向きに同じ高さにある場合、左側にあるパラメーターが先に先に来る引数として処理されます。
- 仮に 2 つのパラメーターが全く同じ位置にある場合、順序の決定に内部的には指定されているコンポーネント ID が自動的に使用されます。



メモ：

- コンポーネントの位置ならびに大きさの変更は「やり直し不可」なアクションです。

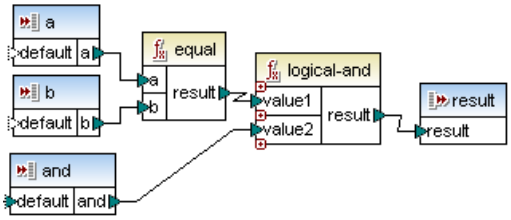
- 新たに追加された入力ならびに出力コンポーネントは、最後の入力または出力コンポーネントの下部にて作成されます。
- 複合型と単純型のパラメーターを混合することもできます。パラメーターの順序は、コンポーネントの位置により決定されます。

8.2.1 関数のパラメーター

関数パラメーターは、**入力**ならびに**出力コンポーネント**としてユーザー定義関数内にて表示されます

入力コンポーネント/パラメーター：**a** ならびに **b**

出力コンポーネント/パラメーター：**result**

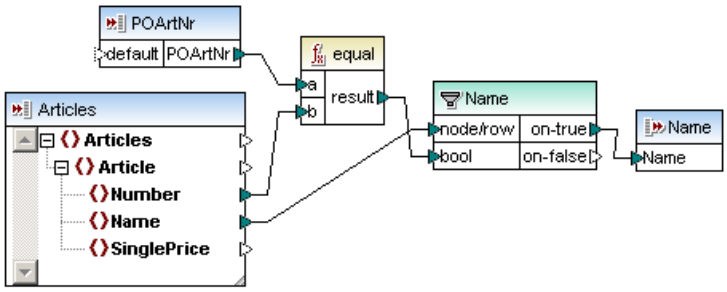


入力パラメーターはメインマッピングからユーザー定義関数へ値を受け渡すために使用され、出力パラメーターはユーザー定義関数からメインマッピングへ値を返すために使用されます。ユーザー定義関数は、他のユーザー定義関数からも呼び出すことができるという点に留意してください。

単純型のパラメーターと複合型のパラメーター

ユーザー定義関数の入力ならびに出力パラメーターは異なる型で定義することができます：

- 単純型の値（例：string や integer）
- 複合型のノードツリー（例：属性や子ノードを含むXML 要素）



入力パラメーターの**POArtNr** では、"string" 単純型の値が受け渡されます。

入力パラメーターの**Articles** は、複合型のXML ドキュメントノードツリーです。

出力パラメーターの**Name** はstring の単純型です

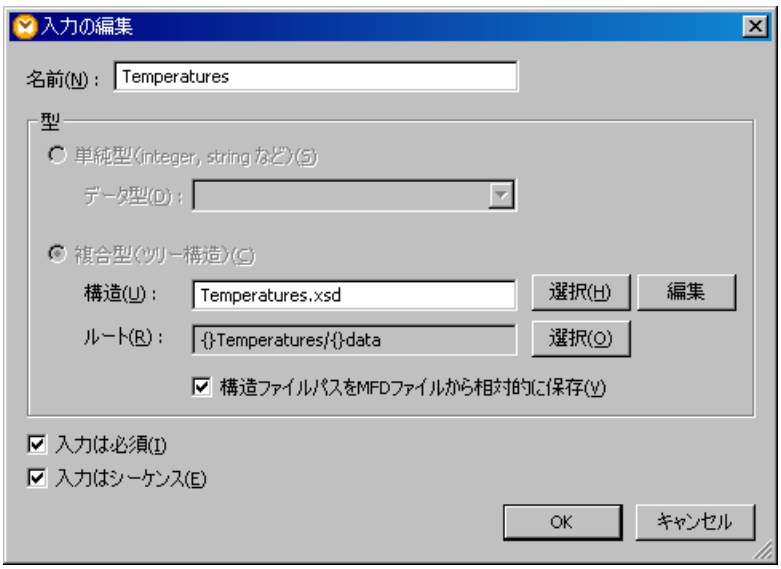
メモ:
上に示されるユーザー定義関数は、..\MapForceExamples フォルダー内に収められている **PersonListByBranchOffice.mfd** ファイルにより確認することができます。

シーケンス

シーケンスは値の範囲またはシーケンスにより構成されるデータのことです。コンポーネントパラメーターダイアログボックスでは、ユーザー定義された単純ならびに複合型の***パラメーター***（入力と出力）をシーケンスとして定義することができます。

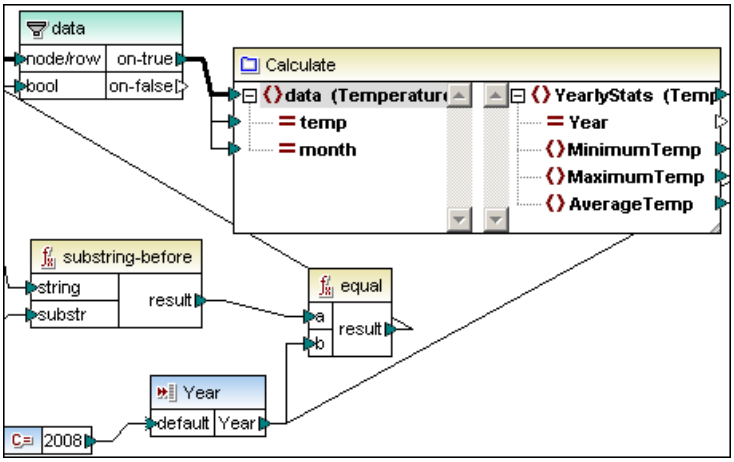
min や max、avg といった集計関数とこのような種類の入力を組み合わせることで、入力シーケンスから特定の値を 1 つ得ることができます。

[入力はシーケンス] チェックボックスが有効になっている場合、コンポーネントは入力をシーケンスとして扱います。このチェックボックスが無効になっている場合、入力は単一の値として扱われます。

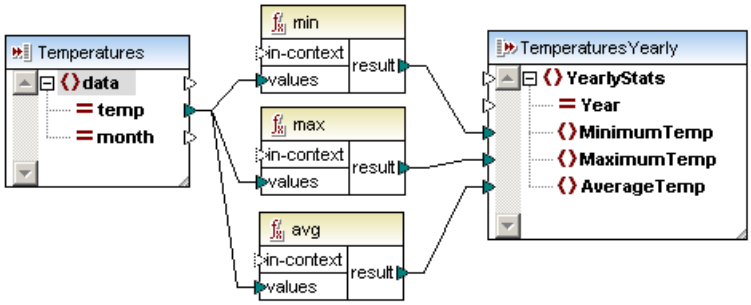


入力データの種類がシーケンスかそうでないかにより、関数が呼ばれる**回数**が決定されます。

- **シーケンス**パラメーターへの接続が作成されている場合、ユーザー定義関数への呼び出しが**一度**だけ行われ、シーケンス全体がユーザー定義関数へ渡されます。



スクリーンショットには、..\MapForceExamples フォルダーに収められている "**InputIsSequence.mfd**" マッピングに含まれているユーザー定義関数の "Calculate" が示されています。以下に示される **Temperatures** 入力コンポーネントはシーケンスとして定義されています。



- **非シーケンス**パラメーターとして接続が行われている場合、シーケンス内にある各アイテムごとに、ユーザー定義関数が**一度**呼ばれます。

メモ:
ユーザー定義関数がインライン型として定義されている場合、入力/出力パラメーターのシーケンス設定は無視されます。

空のシーケンスを非シーケンスパラメーターへ接続すると、関数は全く呼ばれなかったものとして扱われます！

ソースの構造が**オプション**のアイテムであるか、フィルターの条件によりマッチしたアイテムが得られなかった場合に、上に記述したような状況が起こりえます。このような状況を回避するには、関数への入力を行う前にsubstitute-missing 関数を使用してシーケンスが空にならないことを保証するか、パラメーターをシーケンスとしてセットし、関数内部でシーケンスが空だった時の処理を追加する必要があります。

関数により複数の値が出力コンポーネントへ与えられ、出力コンポーネントがシーケンスとして定義されていない場合、最初の結果だけが関数が呼ばれた際に使用されます。

8.2.2 インラインと一般的なユーザー定義関数

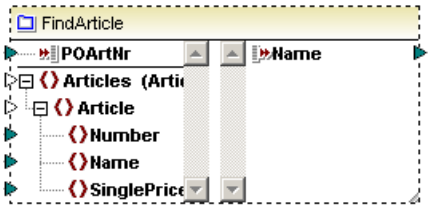
インライン関数はコードが生成された際に実装されるという点で、通常の関数とは根本的に異なるものです。

- **インライン**型関数のコードは、ユーザー定義された関数が呼び出される/使用される**全ての場所**に挿入されます。
- **通常**関数のコードは、関数呼び出しとして実装されます。

  従って、インライン関数は関数内部の実装により置き換えられたかのような振る舞いをします。この仕様は、複雑なマッピングを小さな内部パーツに置き換える際に適したものです。

メモ:
インライン関数を使用することで、生成されたプログラムコードの量が飛躍的に増加します！ユーザー定義関数に対応するコードが、関数が呼び出される/使用される全ての箇所に挿入され、通常の関数を使用するのに比べてコードの大きさが飛躍的に増加します。

インラインユーザー定義関数は破線の境界により表示されます：



インラインユーザー定義関数では、以下が**サポート**されます：

- 関数内部における**複数の出力コンポーネント**

以下はサポートされません：

- パラメーターに対する優先コンテキストの設定
- インラインユーザー定義関数に対する再帰的な呼び出し

一般的な ユーザー定義関数 (**実線**の境界線で表示される非インライン関数)：



一般的な (非インライン) ユーザー定義関数では以下の項目が**サポート**されます:

- 関数内における**単一の出力コンポーネント**
- **再帰的な**呼び出し (再帰を終了する分岐に接続されているIf-Else 条件を与えるなどして、終了条件を指定する必要があります)
- 優先コンテキストパラメーター

メモ:

通常の関数において複数の出力コンポーネントは**サポートされません**が、**作成することはできます**。しかし、コードの生成やマッピング結果のプレビューを行おうとする際にエラーメッセージが表示されます。

関数内で再帰を使用しない場合、関数の種類を「インライン」に変更することができます。

以下の項目はサポートされません：

- 非シーケンスの**入力**コンポーネントからフィルターコンポーネントへの直接接続
- シーケンスや、単一の入力コンポーネントから接続された集計関数 (例：exists、substitute-missing、sum、group-by)

コード生成

一般的なユーザー定義関数の実装は、呼び出し可能なXSLT テンプレートまたは関数として一度生成されます。入力がパラメーターとして渡され、出力が関数 (コンポーネント) の戻り値となっている箇所では、対応するユーザー定義関数コンポーネントの**関数呼び出し**コードが生成されます。

ランタイムにおいて、全ての入力パラメーター値は最初に評価され、各入力データに対して関数の呼び出しが行われます。この処理の詳しい情報については関数パラメーターのセクションを参照ください。

ユーザー定義関数の「種類」を変更する：

1. ユーザー定義関数をダブルクリックして、関数を構成するコンポーネントを確認します。
2. メニューオプションから「関数 | 関数設定」を選択し、「インラインを使用」チェックボックスにチェックを入れます。

ユーザー定義関数と全てコピー接続

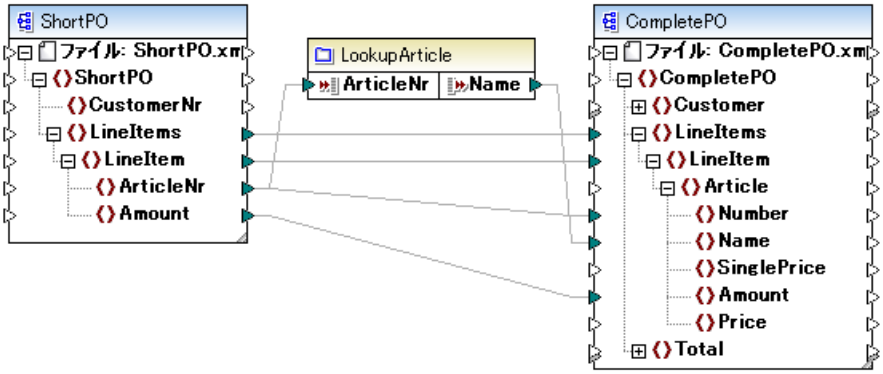
スキーマを複合型のユーザー定義関数パラメーター間にて全てコピー接続を作成する場合、両方のコンポーネントが同一のスキーマをベースにしたものである必要があります！同一のルート要素である必要はありません。複合型出力コンポーネント一定義 のセクションからサンプルを確認することができます。

8.2.3 単純型のルックアップ関数を作成する

以下のサンプルは ...\ MapForceExamples フォルダーに収められている **lookup- standard.mfd** ファイルにて確認することができます。

目的：
以下にあるような、一般的なルックアップ関数を作成する：

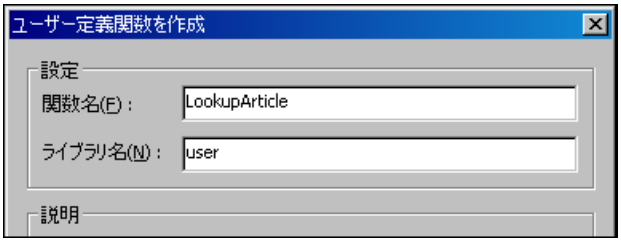
- Articles XML ファイルから得られるArticles/Number データを、別のXML ファイル（この例ではShortPO）にあるArticle の番号と比較します。



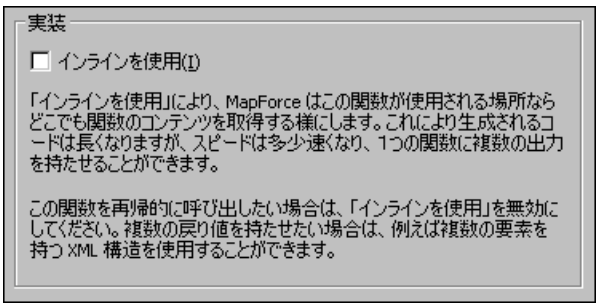
- ShortPO.xsd を挿入し、ソースXML ファイルとしてShortPO.xml を割り当てます。
- CompletePO.xsd スキーマファイルを挿入し、ルート要素としてCompletePO を選択します。
- 以下に記述されている方法により、新たなユーザー定義関数を挿入します。

ユーザー定義関数の作成：

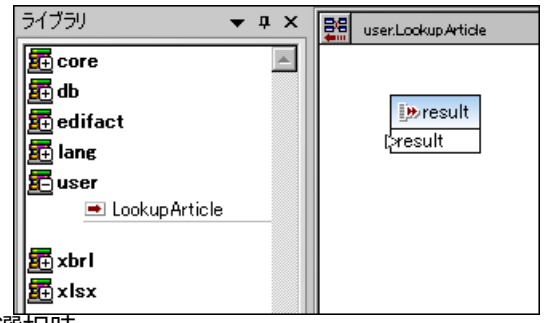
1. メニューオプションから「関数 | ユーザー定義関数の作成」を選択します。
2. 関数の名前を入力します（例：LookupArticle）。



3. 「インラインを使用」チェックボックスのチェックを外し、「OK」ボタンをクリックして確定します。



"result" という名前の出力コンポーネントだけがタブの中に表示されます。

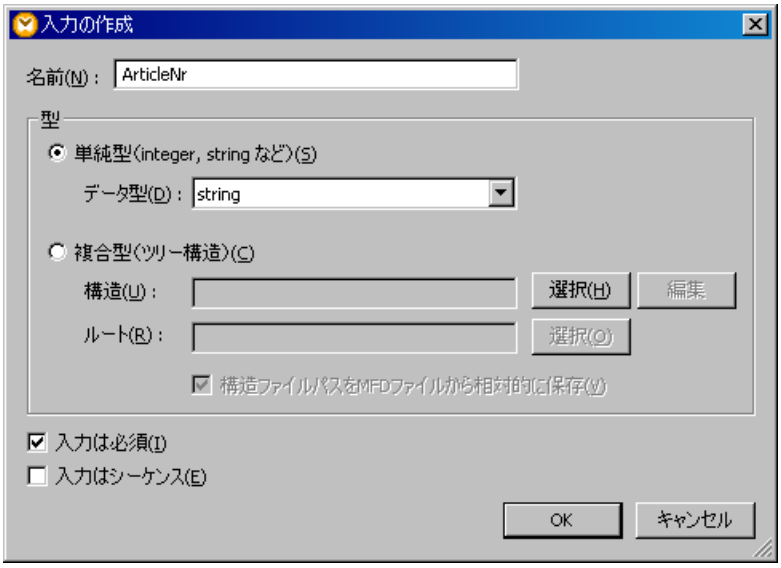


Java 選択時

これがユーザー定義関数を定義するための作業エリアになります。

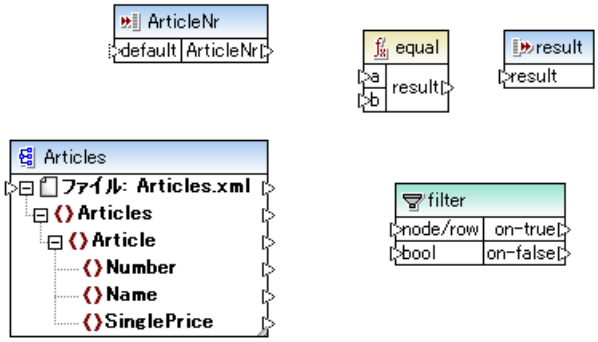
"user" という名前のライブラリが、ライブラリペインにて新たに作成され、"LookupArticle" という名前の関数がその下に表示されます。

4. **スキーマ/XML ファイルの挿入** アイコン をクリックすることで、**Articles** スキーマを挿入し、同名の XML ファイルをデータソースとして選択します。
5. **入力挿入** アイコン をクリックして、入力コンポーネントを挿入します。
6. 入力パラメーターの名前（この例では ArticleNr）を入力し、**「OK」** をクリックします。



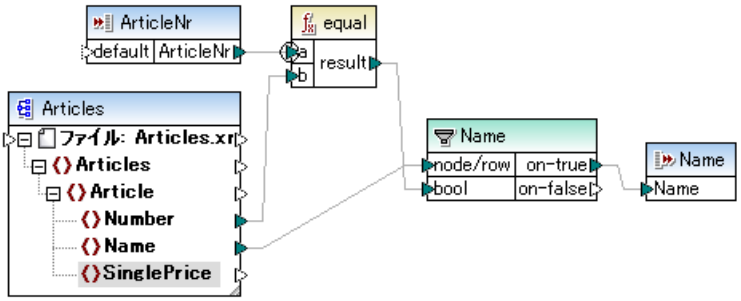
ユーザー定義関数の入力として機能するユーザー定義関数の入力アイコンが表示されます。

7. core ライブラリの logical 関数グループから **"equal"** コンポーネントをドラッグすることで、新たな関数を挿入します。
8. フィルターの挿入アイコン をクリックして、**フィルター** コンポーネントを挿入します。



以下に示されているマッピングを参考に、ユーザー定義関数のマッピングを行います。以下の点に注意してください：

9. **a** パラメーターを右クリックして、コンテキストメニューから**優先コンテキスト**を選択します。
10. 出力コンポーネントを**ダブルクリック**して、出力パラメーターの名前を変更します（この例では、"**Name**" に変更します）。



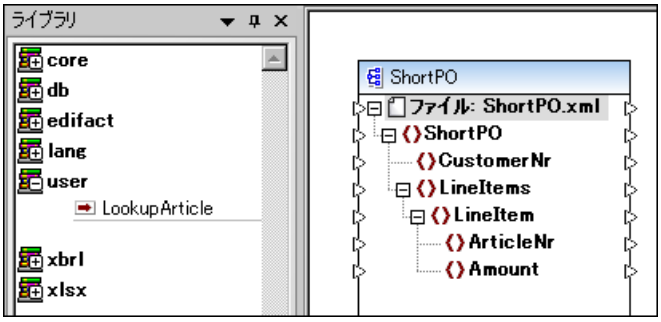
これでユーザー定義関数の定義が完了しました。

メモ：
入力または出力コンポーネントをダブルクリックすることでダイアログボックスが表示され、入力パラメーターの名前やデータ型を変更したり、関数に入力アイコンが表示されるか（入力は必須）、また入力をシーケンスとして扱うかを指定することができます。

このユーザー定義関数は、以下のような特徴を持ちます：

- ShortPO XML ファイルからデータを受け取るArticleNr という名前の**入力**を 1 つ備えている。
- ShortPO ArticleNr の値と、（この目的のために関数に挿入されている）Articles XML インスタンスファイルから得られたArticle/Number を**比較**する。
- **フィルター**コンポーネントを使用することで、比較の結果がtrue の場合にArticle/Name レコードを出力コンポーネントへ渡す。
- Name という名前の出力コンポーネントを 1 つ備えており、Article Name レコードをCompletePO XML ファイルへ渡す。

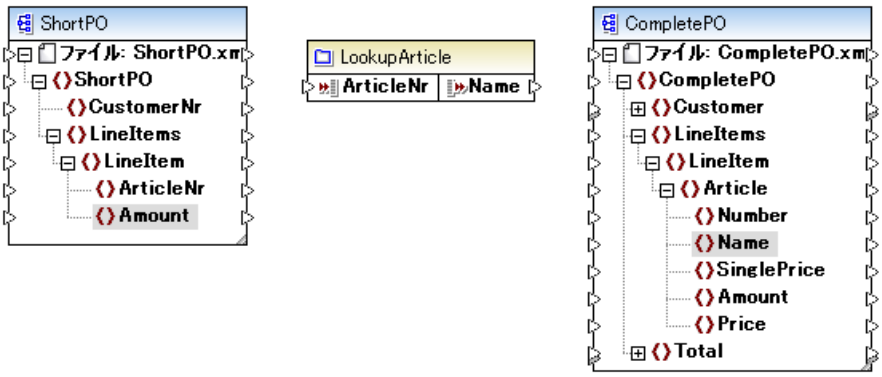
11. ホームアイコン をクリックしてメインマッピングへ戻ります。
ユーザー定義関数のLookupArticle が、user ライブラリ以下に表示されます。



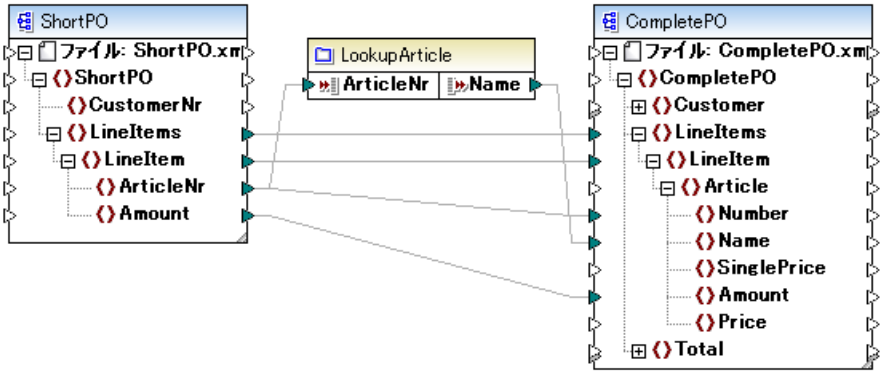
12. LookupArticle 関数をマッピングウィンドウへドラッグします。

ユーザー定義関数が以下のように表示されます。

- タイトル関数バーに、関数名の "LookupArticle" が表示されます。
- 入力ならびに出力アイコンが、それぞれの名前とともに表示されます。



13. 以下のスクリーンショットに示されるようにマッピングを作成し、出力タブをクリックしてマッピングの結果を確認します。

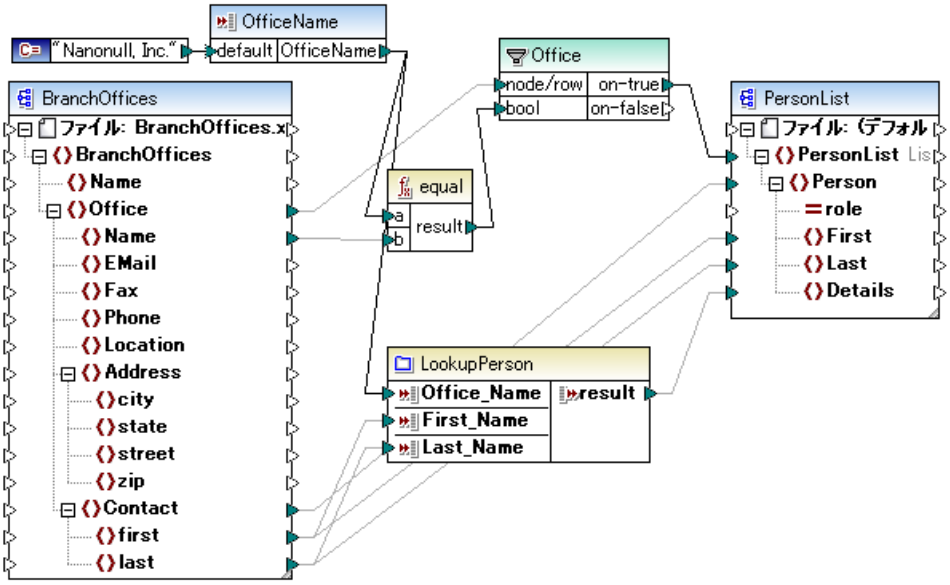


8.2.4 ユーザー定義関数 - サンプル

<Documents>\ Altova\ MapForce2017\ MapForceExamples フォルダーに収められている **PersonListByBranchOffice.mfd** ファイルを例にして、以下にある機能の詳細を解説します：

- ネストされたユーザー定義関数（例：**LookupPerson**）
- 文字列の出力を生成するルックアップ関数（例：**LookupPerson**）
- **デフォルト**値が与えられている**オプション**の入力パラメーター（例：LookupPerson コンポーネントに含まれる **EqualAnd** コンポーネント）

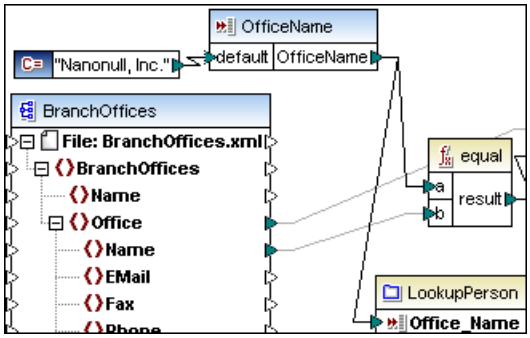
- 生成されたマッピングコードを実行する際に、コマンドラインの引数としても使用することができる入力パラメーター



構成することのできる入力パラメーター

入力パラメーター (OfficeName) では、以下の方法により、マッピングが実行された際にデータを渡すことができます：

- 生成されたコードを実行する際に、**コマンドライン**の引数として与える (例：Mapping.exe /OfficeName "Nanonull Partners, Inc.")。
- 内蔵の実行エンジンを使って出力ウィンドウでデータのプレビューをする際に、**プレビュー**値として与える。

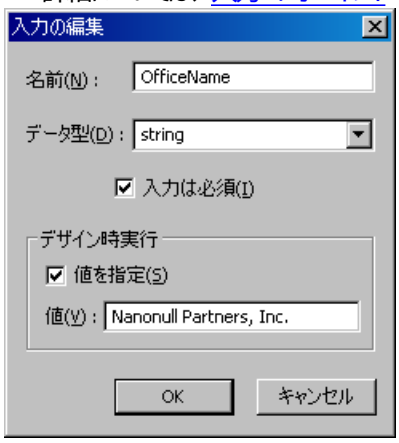


入力値を定義する：

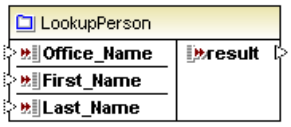
1. 入力コンポーネントをダブルクリックして、デザイン時実行グループにある「値」テキストボックスにて別の値 (例："Nanonull Partners, Inc.") を入力し、**「OK」**をクリックして確定します。
2. 出力タブをクリックして、マッピングの結果を確認します。
   従業員の一覧が表示されます。

   このダイアログボックスで入力されたデータは、**「プレビュー」**モードで (つまり出力タブをクリックした際に) のみ使用される点に注意してください。値が入力されなかった場合、または「値を使用」チェックボックスが無効になっている場合、入力アイコンへのデータマッピングが使用されます。

詳細については、入力コンポーネントを参照ください。



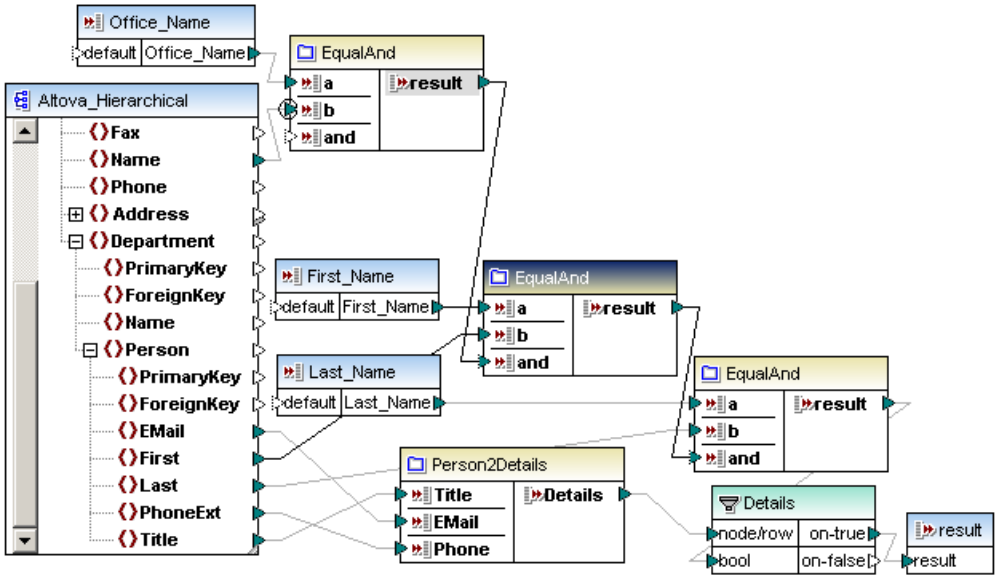
LookupPerson コンポーネント



ユーザー定義関数のコンポーネントをダブルクリックすることで、以下に示されるコンポーネントが表示されます。このコンポーネントは以下のように動作します：

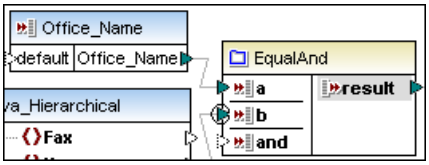
- **入力**コンポーネントと**EqualAnd** ユーザー定義関数コンポーネントにより、Office とBranchOffice.xml のFirst ならびにLast 名が、Altova_Hierarchical.xml ファイルにある同じ名前のフィールドと**比較**されます。
- **Person2Details** ユーザー定義関数を使用することで、Email、PhoneExt、Title アイテムが**連結**されます。
- EqualAnd から得られた結果が全てtrue の場合、処理された従業員のデータを**出力**コンポーネントへ**受け渡し**ます（フィルターコンポーネントにてture かどうかの判定が行われます）。

ユーザー定義関数は、（極論すれば空の文字列でも）常に何らかの値を出力します！この例では、フィルターコンポーネントのbool 値がfalse となった場合、Peson2Details コンポーネントの代わりに空の文字列が出力されます。



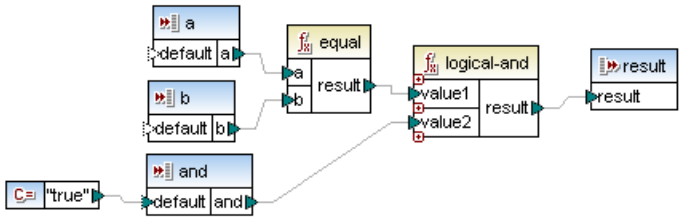
- 3 つの**入力**コンポーネントであるOffice_Name、First_Name、Last_Name によりBranchOffices.xml ファイルからのデータが受け取られます。
- **EqualAnd** コンポーネントにより 2 つの値が比較され、更にデフォルト値を持った**オプション**の比較値が与えられます。
- Person2Details コンポーネントにより、3 つのフィールドが組み合わされ、その結果がフィルターコンポーネントへ渡されます。

EqualAnd コンポーネント



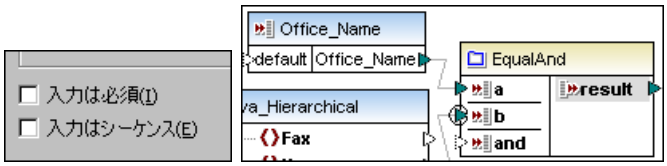
ユーザー定義関数のコンポーネントをダブルクリックすることで、以下に示されるコンポーネントが表示されます。このコンポーネントは以下のように動作します：

- **a** と**b** という 2 つの入力パラメーターを比較し、その結果をlogical-and コンポーネントへ渡します。**b** パラメーターが優先コンテキストとして定義されていることに注目してください（アイコンを右クリックして指定することができます）。これにより、入力パラメーター**a** により与えられた特定オフィスの従業員データが最初に処理されるようになります。
- 最初の比較結果と、オプションの入力パラメーターである "and" を**logical-and** により比較します。
- 最終的にlogical-and から得られた比較結果のboolean 値が出力パラメーターへ渡されます。



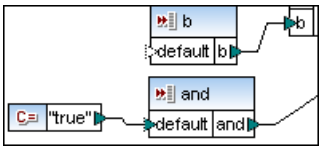
オプションのパラメーター

上に示されるEqualAnd ユーザー定義関数の "and" パラメーターをダブルクリックして、「入力は必須」チェックボックスのチェックを外すことで、そのパラメーターをオプションと指定することができます。



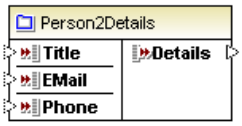
「入力は必須」チェックボックスの**チェックが外されている**場合：

- このユーザー定義関数の入力アイコンに対して、マッピングを行う必要が無くなります。例えば、LookupPerson 関数内にある最初のEqualAnd 関数の**and** 入力パラメーターはマッピングされていません。
- ユーザー定義関数内部にあるコンポーネントからマッピングを行うことで、**デフォルト**値を与えることができます。例えば、"true" という値を含む定数コンポーネントを使用することができます。



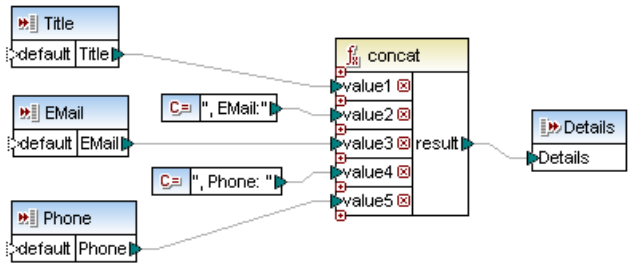
- 他のアイテムからオプションとなっている入力コンポーネントへマッピングを行うことで、デフォルト値の値がオーバーライドされます。例えば、LookupPerson 関数内部にある 2番目のEqualAnd 関数には、最初のEqualAnd ユーザー定義関数の "result" パラメーターの値が与えられます。

Person2Details コンポーネント



このユーザー定義関数コンポーネントをダブルクリックすることで、以下に示されるコンポーネントが表示されます。このコンポーネントは以下のように動作します：

- 3 つの入力を連結して、result 文字列を出力パラメーターへ渡します。
- 出力パラメーターをダブルクリックすることで、パラメーター名を変更したり (Details)、データ型を選択することができます (string)。



8.2.5 複合型のユーザー定義関数 - XML ノードを入力に使用

以下のサンプルは…\MapForceExamples フォルダーに収められている**lookup-udf-out.mfd** ファイルにて確認することができます。このセクションでは、複合型の入力コンポーネントを持つインラインのユーザー定義関数の作成方法について説明します。

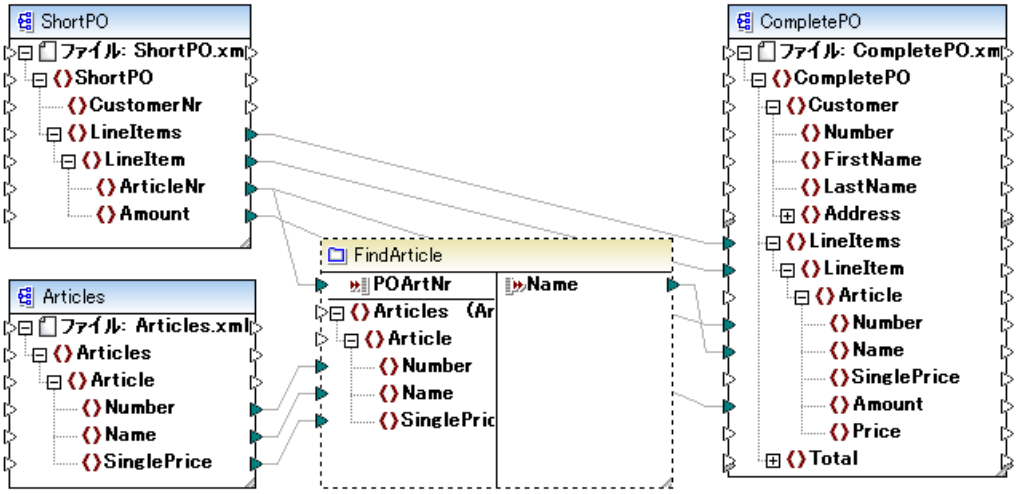
ユーザー定義関数の "FindArticle" は 2 つのパーツから構成されることに注意してください。

以下の入力パラメーターを含む左半分：

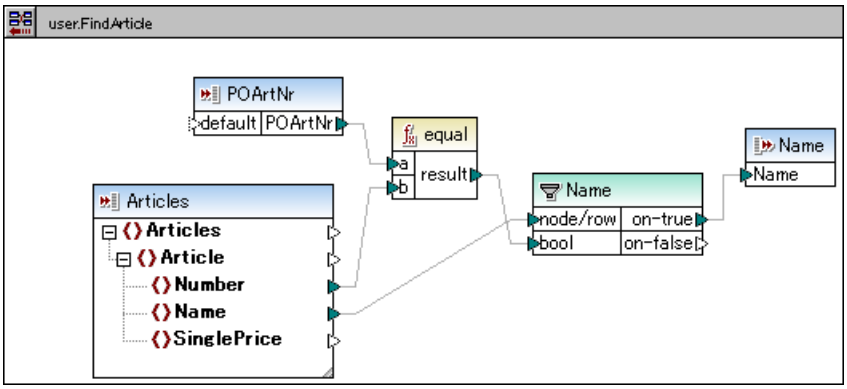
- 単純型の入力パラメーター：**POArtNr**
- XML 子ノードへ直接マッピングする複合型入力コンポーネント **Articles**

以下のコンポーネントを含む右半分：

- **"Name"** という名前を持つ単純型出力パラメーター



.ユーザー定義関数を行使絵するコンポーネントを以下のスクリーンショットに示します。左側に2つの入力コンポーネントがあり、右側には出力コンポーネントが表示されています

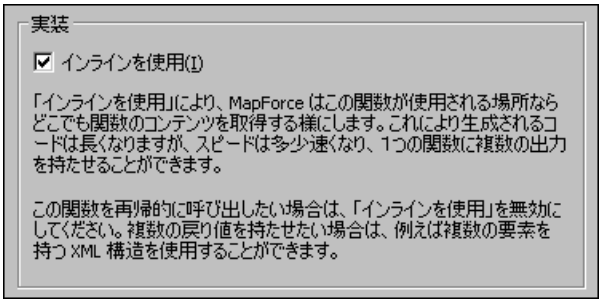


8.2.5.1 *複合型の入力コンポーネント - 定義*

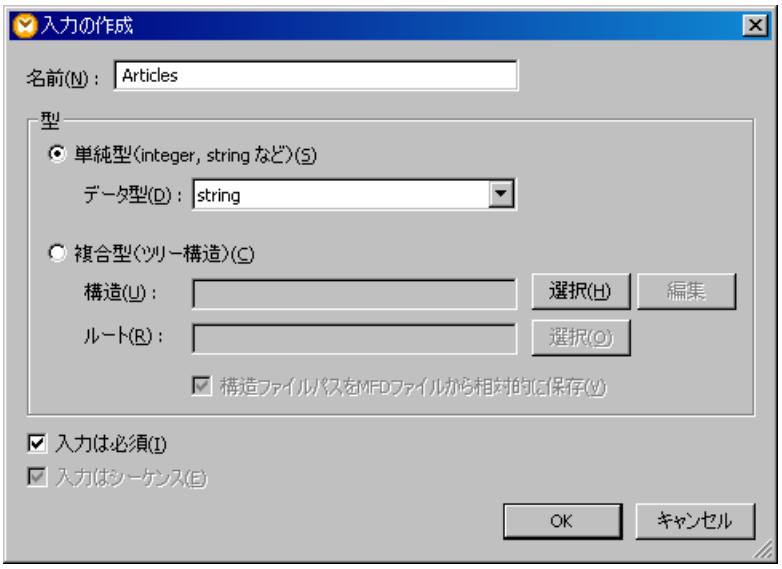
XML 構造を入力パラメーターとする関数の作成方法は、以下のとおりです：

1. 通常の方法でユーザー定義関数を作成します。メニューオプションから 関数 | ユーザー定義関数の作成」を選択

し、**「OK」**をクリックすることで確定します。インラインを使用チェックボックスが自動的に選択されていることに注目してください。



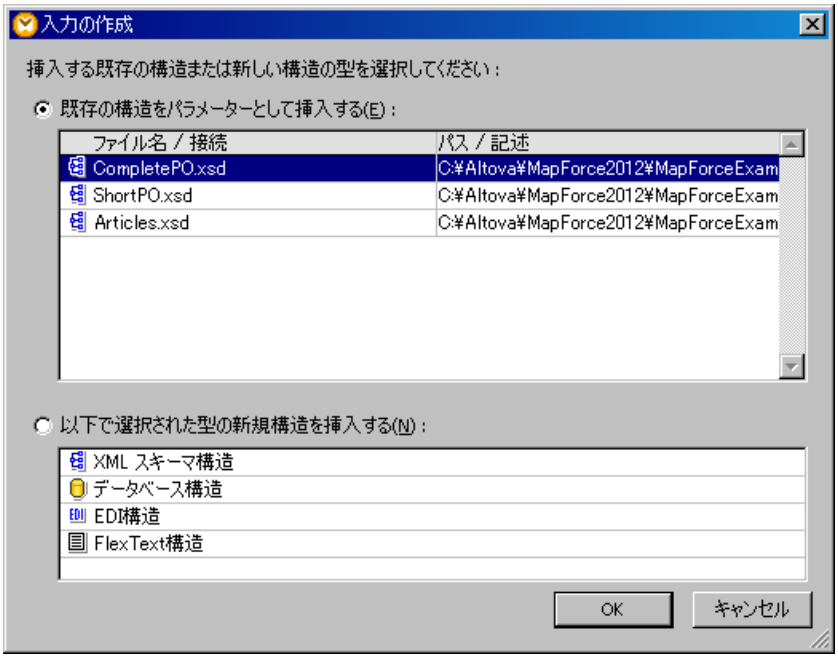
2. アイコンバーにある**入力挿入**アイコン  をクリックします。
3. 入力コンポーネントの名前を名前フィールドに入力します。



4. **複合型（ツリー構造）**ラジオボタンをクリックして、構造フィールドの隣にある**選択**ボタンをクリックします。これによりダイアログボックスが開かれます。

   上部のリストボックスには、マッピングにおける既存のコンポーネントが表示されます（サンプルファイルを開いた場合、3 つのスキーマが表示されます）。このリストには、アクティブなマッピングに挿入された全てのコンポーネント（XML スキーマ、データベース、EDI ファイルまたは FlexText 構造 ファイル）。

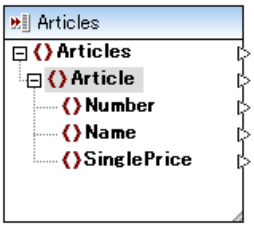
   下部のリストボックスでは、複合型のデータ構造を新たに選択することができます（例：XML スキーマ、データベース、EDI ファイル、または FlexText 構造ファイル）。



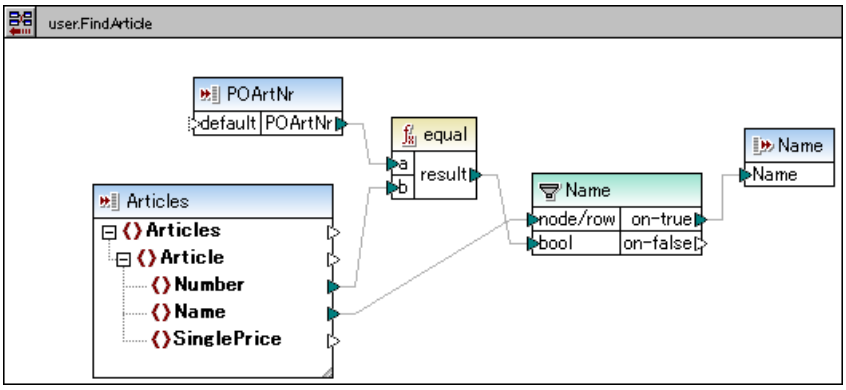
5. 「新規構造を挿入する」ラジオボタンを選択し、XML スキーマ構造エントリーを選択した後、**「OK」**ボタンをクリックします。
6. ダイアログボックスから**Articles.xsd** を選択します。
7. コンポーネントのルート要素として使用する要素（例：Articles）をクリックした後、**「OK」**をクリックします。



Articles コンポーネントがユーザー定義関数へ挿入されます。コンポーネント名の左側に入力アイコン が表示されていることに注目してください。コンポーネントが複合型入力コンポーネントとして使用されることを表しています。



8. 以下のスクリーンショットに示されるように、その他のコンポーネントを挿入します。POArtNr という名前の 2 番目の入力コンポーネント、フィルターコンポーネント、equal 関数を挿入し、出力コンポーネントの名前を Name に変更します。以下のスクリーンショットに示されるように接続を作成します。

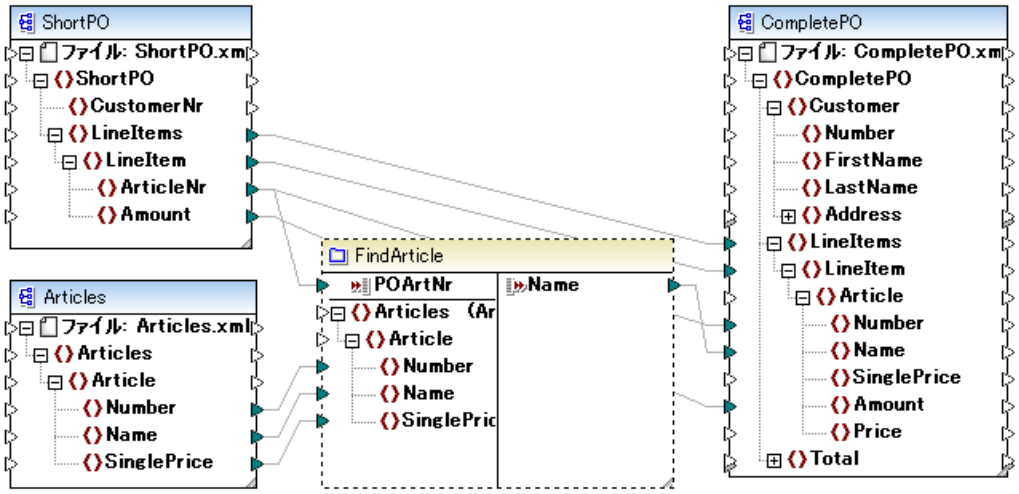


- **Articles** 入力コンポーネントは、ユーザー定義関数の**外から**データを受け取ります。このコンポーネントに対するマッピングは、ユーザー定義関数の外側で表示される入力アイコンから行うことができます。
- データを与えるために使用される XML **インスタンス**ファイルを、ユーザー定義関数の内部にて複合型入力コンポーネントに割り当てることはできません。
- 別の入力コンポーネント POArtNr により、ShortPO ArticleNr のデータが与えられ、**Article | Number** のデータと比較されます。
- フィルターコンポーネントにより、同一のレコードがフィルタリングされ、出力コンポーネントに渡されます。

9. ホームアイコン をクリックして、メインマッピングに戻ります。
10. 新たに作成されたユーザー定義関数をライブラリペインからマッピングへドラッグします。



11. 以下のスクリーンショットに示されるような接続を作成します。



マッピングの左側では、2つのスキーマ/XML ファイルのアイテムからマッピングされる入力パラメーターが示されます：

- **ShortPO** により、入力コンポーネントの**POArtNr** に対するデータが与えられます。
- **Articles** からは、複合型の入力コンポーネントに対するデータが与えられます。Articles スキーマファイルのコンポーネントが挿入された際にArticles.xml インスタンスファイルが割り当てられました。
- XML 子ノードを伴う複合型入力コンポーネントのe complex 入力コンポーネント**Articles** には、Articles コンポーネントからデータがマッピングされます。

マッピングの右側には、以下のアイテムが表示されます：

- **"Name"** という名前の単純型出力パラメーターでは、同じArticle 番号によりフィルタリングされたアイテムが与えられ、CompletePO のName アイテムへ渡されます

```
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>
```

メモ：
スキーマとユーザー定義関数のパラメーター間にて、**全てコピー**接続を作成する場合、両方のコンポーネントが同じスキーマをベースにしている必要があります！両コンポーネントが同一のルート要素を持っている必要はありません。

8.2.6 複合型のユーザー定義関数 - XML ノードを出力に使用

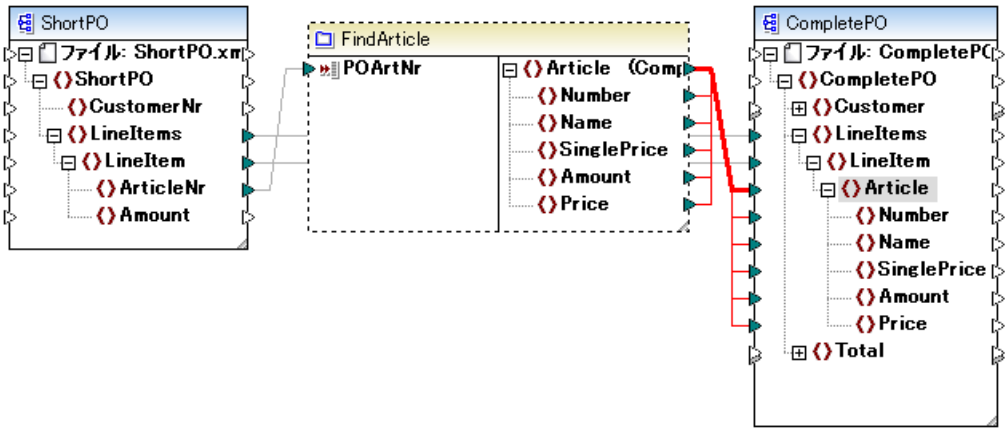
以下のサンプルは**...\ MapForceExamples** フォルダーに収められている**lookup-udf-out.mfd** ファイルにて確認することができます。このセクションでは、複合型の出力コンポーネントを持つインラインのユーザー定義関数の作成方法について説明します。
ユーザー定義関数の "FindArticle" は 2つのパーツから構成されることに注意してください。

以下の入力パラメーターを含む左半分：

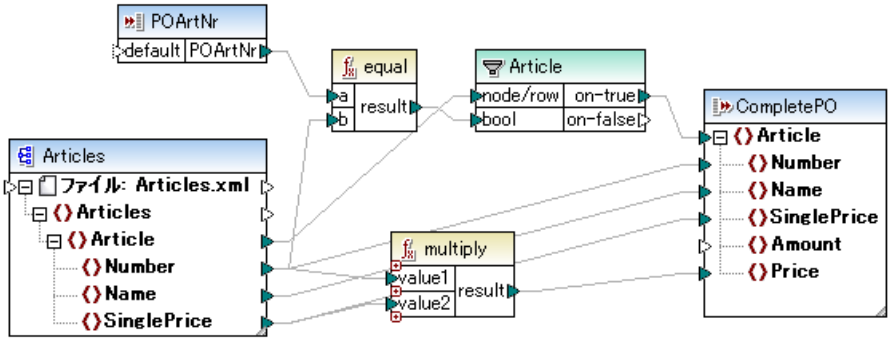
- 単純型の入力パラメーター：**POArtNr**

以下のコンポーネントを含む右半分：

- CompletePO へマッピングされたXML 子ノードを伴う複合型の出力パラメーター：**Article (CompletePO)**



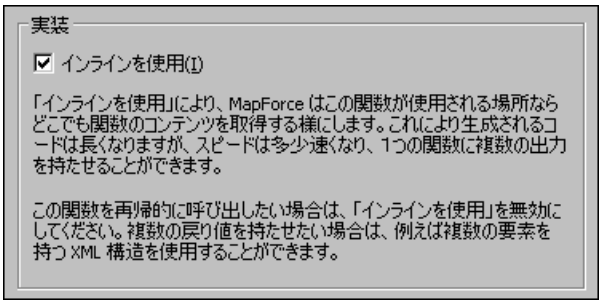
ユーザー定義関数を行使絵するコンポーネントを以下のスクリーンショットに示します。左側に入力コンポーネントがあり、右側には複合型の出力コンポーネントが表示されています。



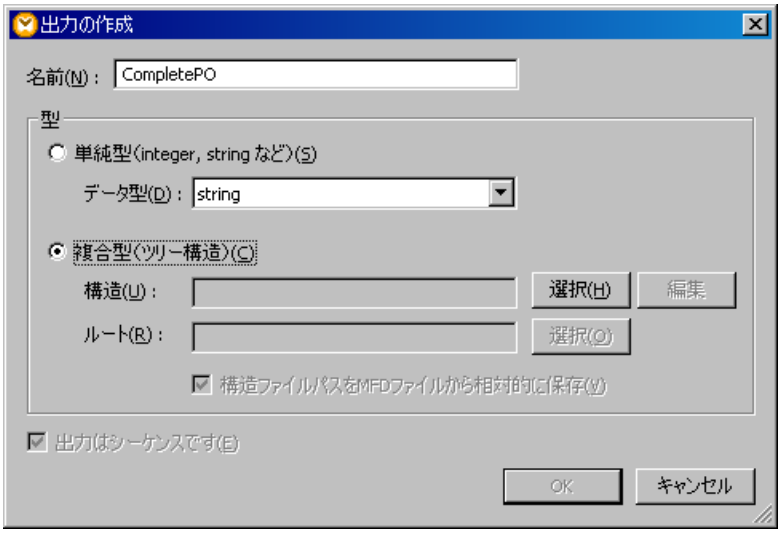
8.2.6.1 *複合型の出力コンポーネント - 定義*

XML 構造を出力パラメーターとして返す関数の作成方法は、以下のとおりです：

1. 通常の方法でユーザー定義関数を作成します。メニューオプションから 関数 | ユーザー定義関数の作成」を選択し、FindArticle という関数名を与え、 **「OK」** をクリックすることで確定します。**インライン**を使用オプションが自動的に選択されます。



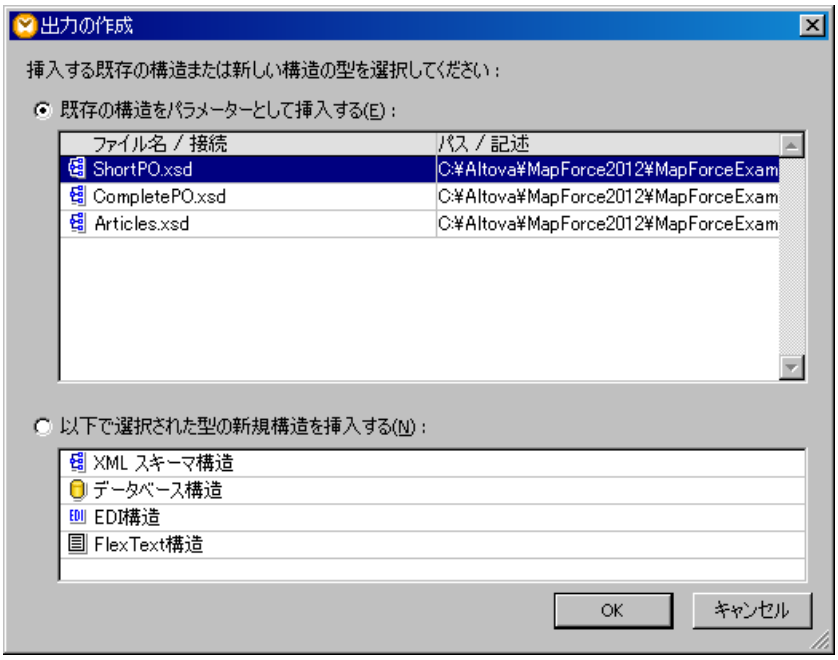
2. アイコンバーにある**出力**アイコン  をクリックして、名前を入力します 例：CompletePO)。



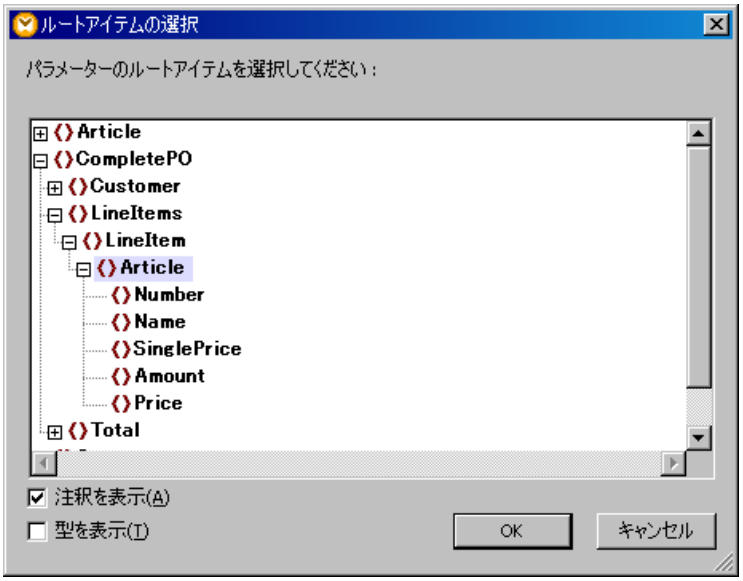
3. **複合型**ラジオボタンをクリックして、構造フィールドの隣にある**選択**ボタンをクリックします。
別のダイアログボックスが新たに表示されます

上部のリストボックスには、マッピングにおける**既存**のコンポーネントが表示されます（サンプルファイルを開いた場合、3 つのスキーマが表示されます）。このリストには、アクティブなマッピングに挿入された全てのコンポーネント（XML スキーマ、データベース、EDI ファイル、FlexText 構造 ファイル）が含まれていることに注目してください。

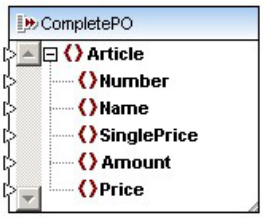
下のリストボックスから、以下の新規の複合型データ構造を選択することができます：XML スキーマ、データベース、EDI ファイル、FlexText 構造 ファイル）。



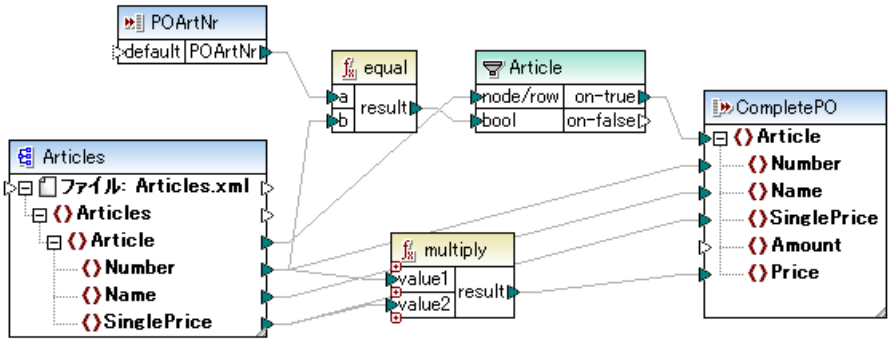
4. "**新規構造の挿入 ...**" ラジオボタンをクリックして、XML スキーマ構造のエントリを選択し、「OK」をクリックして続行します。
5. "開く"ダイアログボックスから**CompletePO.xsd** を選択します。
6. コンポーネント内でルート要素になる要素（例えば、**Article**）をクリックします。「OK」をクリックしてダイアログボックスを閉じます。



CompletePO コンポーネントはユーザー定義関数に挿入されます。出力アイコン がコンポーネント名の左横に表示されることに注意してください。これは、コンポーネントが複雑型出力コンポーネントに使用されていることを示します。



7. Article スキーマ/XML ファイルをユーザー定義関数に挿入し、**Articles.xml** をXML インスタンスとして割り当てます。
8. 残りのコンポーネントを下のスクリーンショットに表示されるように挿入します。つまり：'単純型 "入力コンポーネント (POArtNr)、フィルター、equal、とmultiply コンポーネントが以下のように接続されます。

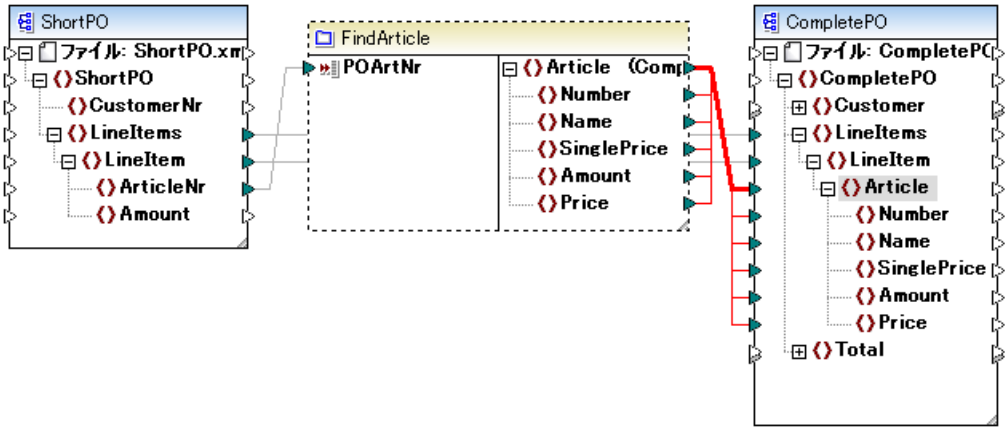


- ユーザー定義関数内部では、**Articles** コンポーネントによりArticles.xml インスタンスファイル内にあるデータが与えられます。
- 入力コンポーネントからはPOArtNr (ArticleNr) が与えられ、Articles | Number とPrice が計算に使用されます。
- フィルターコンポーネントにより、両方の番号が等しいレコードだけがCompletePO 出力コンポーネントへ渡されます。

9. ホームアイコン をクリックして、マッピングへ戻ります。
10. ライブラリペインにて新たに作成されたユーザー定義コンポーネントをマッピングへドラッグします。



11. 以下のスクリーンショットにあるような接続を作成します。
Article (CompletePO) への接続を作成したら、接続の線を右クリックして、コンテキストメニューから「全てコピー」を選択して下さい。その他のコネクタが自動的に生成され、以下のスクリーンショットに示されるように、ハイライトされます。



メモ:
スキーマと「インライン」型のユーザー定義関数間で**全てコピー**接続を作成する場合、両方のコンポーネントが同じスキーマをベースにしている必要があります！両コンポーネントが同一のルート要素を持っている必要はありません。

関数コンポーネントの左側には入力パラメーターが表示されており、単純型のアイテムがマッピングされています。

- ShortPO から**POArtNr** 入力コンポーネントへArticleNr が渡されます。

関数コンポーネントの右側には以下のものが含まれます：

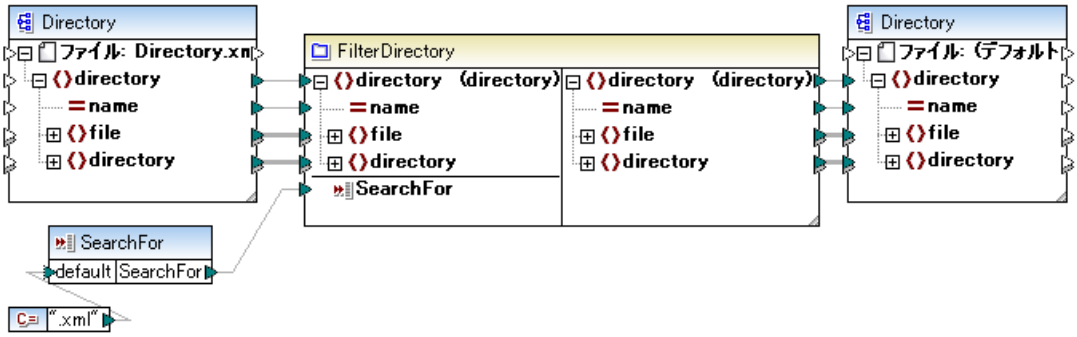
- フィルタリングされたアイテムが与えられる "**Article (CompletePO)**" という名前の複合型出力コンポーネントが、子ノードとともにCompletePO へマッピングされます。

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3    <LineItems>
4      <LineItem>
5        <Article>
6          <Number>3</Number>
7          <Name>Pants</Name>
8          <SinglePrice>34</SinglePrice>
9          <Price>102</Price>
10       </Article>
11     </LineItem>
12     <LineItem>
13       <Article>
14         <Number>1</Number>
15         <Name>T-Shirt</Name>
16         <SinglePrice>25</SinglePrice>
17         <Price>25</Price>
18       </Article>
```

8.2.7 再帰的なユーザー定義マッピング

このセクションでは **..\MapForceExamples** フォルダー内部の **RecursiveDirectoryFilter.mfd** にあるマッピングの作成方法ならびに再帰的なマッピングのデザイン方法について説明します。MapForceExamples プロジェクトフォルダーには、再帰的なマッピングを行う際に参照することのできるサンプルファイルが他にも収められています。

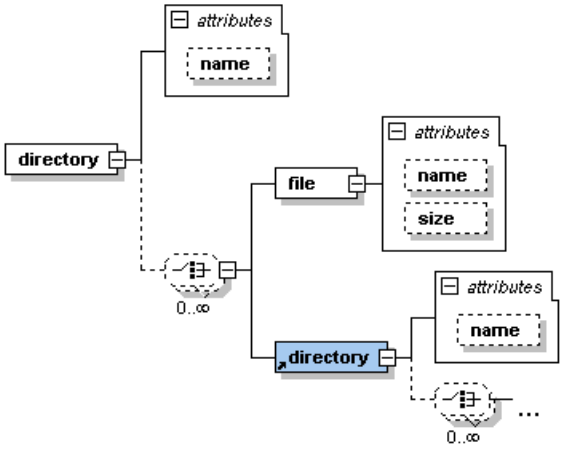
以下のスクリーンショットでは、再帰的なユーザー定義関数の FilterDirectory を含むマッピングが示されており、ソースファイル内にある xml ファイルのリストをフィルタリングします。



このマッピングに対するファイルならびにディレクトリのデータが収められている**ソースファイル**は directory.xml です。この XML ファイルにより、ディレクトリならびにファイルのデータが階層構造で与えられます（以下のスクリーンショットを参照）。

```
<directory name="output">
  <file name="examplesite1.css" size="3174"/>
  <directory name="images">
    <file name="blank.gif" size="88"/>
    <file name="block_file.gif" size="13179"/>
    <file name="block_schema.gif" size="9211"/>
    <file name="nav_file.gif" size="60868"/>
    <file name="nav_schema.gif" size="6002"/>
  </directory>
 </directory>
</directory>
<directory name="Import">
  <file name="altova.mdb" size="266240"/>
  <file name="Data_shape.mdb" size="225280"/>
</directory>
<directory name="IndustryStandards">
  <directory name="News">
    <file name="high-tide.jpg" size="10793"/>
    <file name="Newsml-example.xml" size="5004"/>
    <file name="nitf-example.xml" size="9327"/>
  </directory>
```

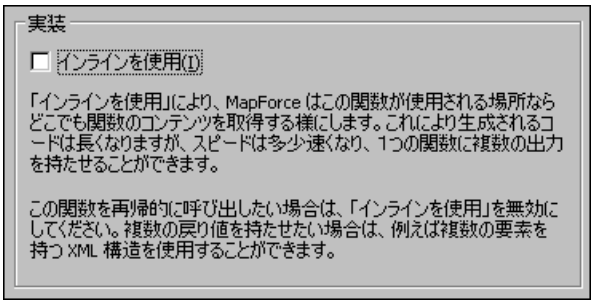
Directory.xml により参照されている XML スキーマファイルには、"directory" という名前の**再帰的**な要素が含まれており、directory 要素以下に任意の数のサブディレクトリを配置することができるようになっています。



8.2.7.1 再帰的なユーザー定義関数の定義

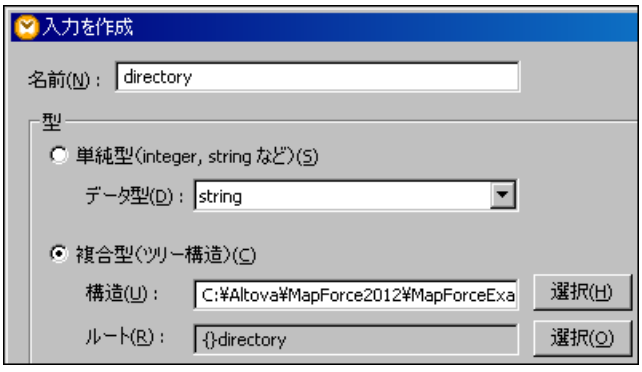
再帰的ユーザー定義関数の作成方法は、以下のとおりです：

1. メニューオプションの「関数 | ユーザー定義関数の作成」を選択して、作成する関数の名前（例えばFilterDirectory）を入力します。
2. 実装グループにある**インラインを使用**チェックボックスのチェックを外して、**「OK」**をクリックしてください。これで関数を再帰的にデザインすることができるようになります。

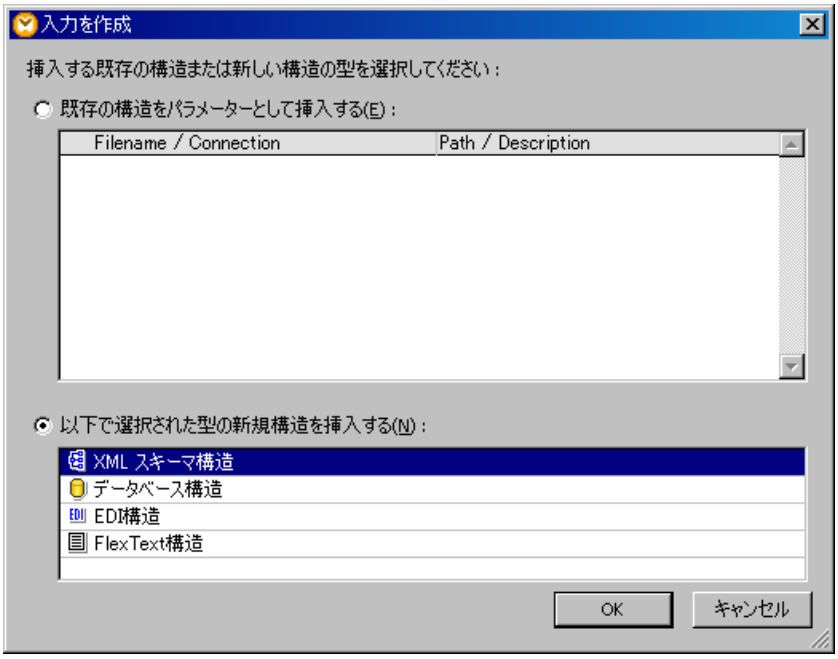


これで**FilterDirectory**ウィンドウが表示され、ユーザー定義関数の作成を始めることができます。

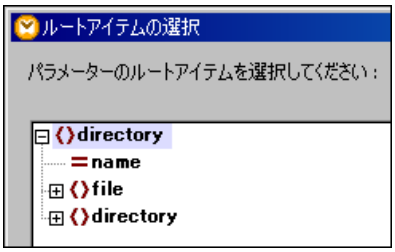
3. メニューから「関数 | 入力の挿入」を選択して、入力コンポーネントを挿入します。
4. 入力コンポーネントの名前（例えばdirectory）を入力し、**複合型**（ツリー構造）ラジオボタンをクリックします。



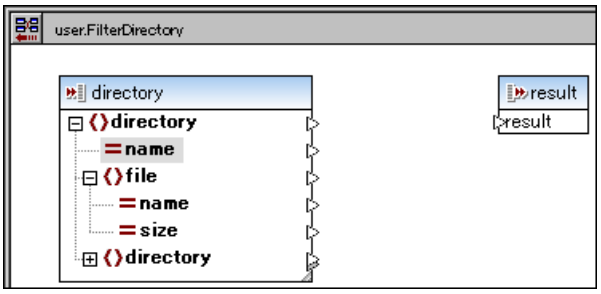
5. 選択ボタンをクリックして、表示される入力を作成ダイアログボックスにてXML スキーマ構造をクリックし、**「OK」**をクリックします。



6. ダイアログボックスにて .\MapForceExamples フォルダーの**Directory.xsd** ファイルを選択します。
7. ルートアイテムを選択するダイアログにて **「OK」** をクリックして、以下に示される様にdirectory を選択します。



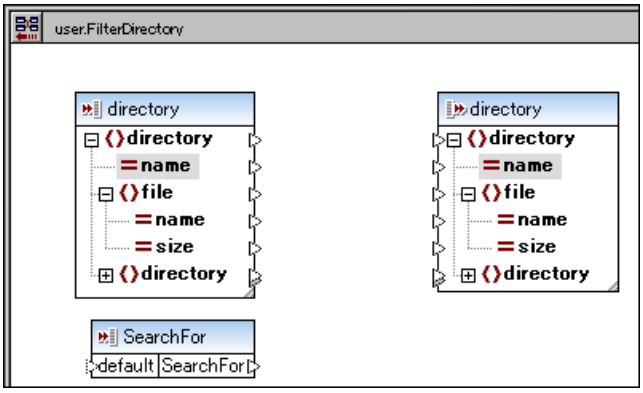
8. **「OK」** クリックして、複合型の入力パラメーターを挿入します。
ユーザー定義関数が表示されます。



9. 複合型の出力コンポーネントが必要であるため、単純型の出力コンポーネントを削除します。
10. 「関数 | 出力の挿入」を選択し、上記にある方法にて出力コンポーネントを挿入します。複合型の出力コンポーネントである "directory" を作成します。

入力側に 1つ、出力側に 1つの複合型コンポーネントが作成されました

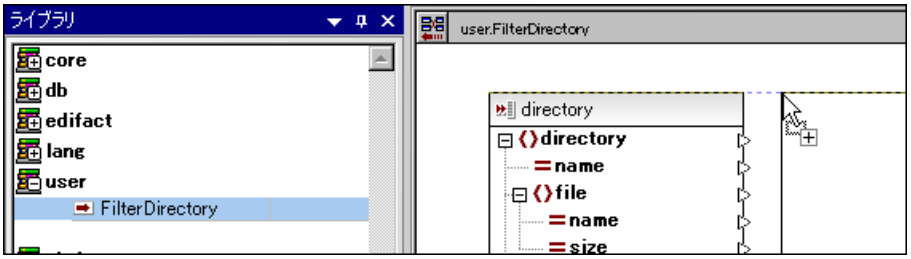
11. 「関数 | 入力の挿入」を選択して、単純型のコンポーネントである**SearchFor** を挿入します



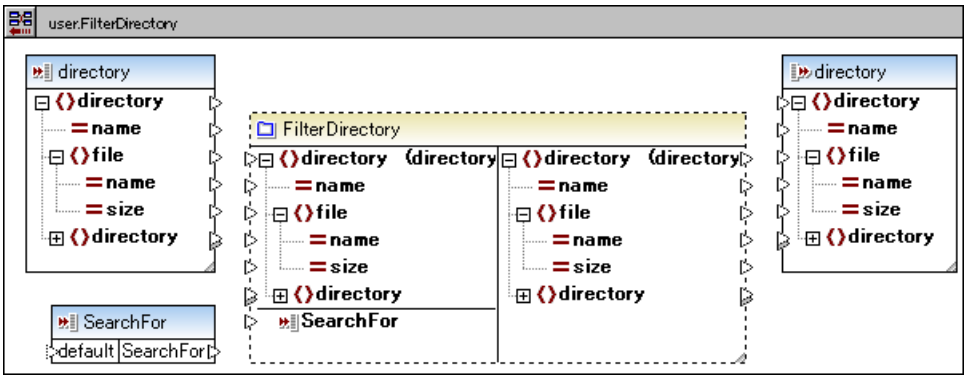
再帰的なユーザー定義関数を挿入する

この時点で必要な入力ならびに出力コンポーネントが全て定義されました。次に「未完成の」関数を、現在のユーザー定義関数ウィンドウに挿入する必要があります（この操作は始どいつでも行うことができます）。

1. **ライブラリ**ウィンドウの**user** セクションにはFilterDirectory 関数があります。
2. FilterDirectory をクリックして、これまで作業してきたFilterDirectory ウィンドウへ関数をドラッグします。

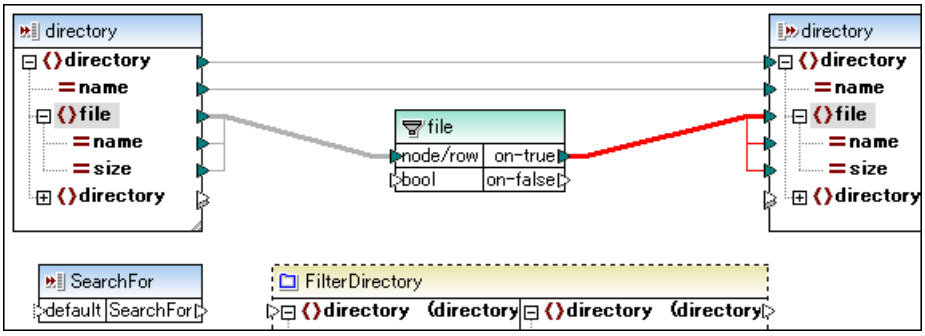


ユーザー定義関数が、再帰的なコンポーネントとしてユーザー定義関数ウィンドウに表示されます。

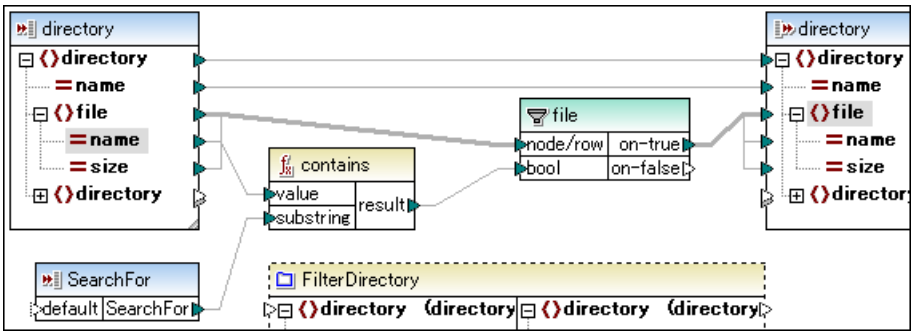


3. 入力コンポーネントの**directory**、**name** そして**ファイル**アイテムを、出力コンポーネントにある同名のアイテムへ接続します。
4. **ファイル**アイテム間で接続されているコネクタを右クリックして、ノード行フィルターを挿入を選択します。
5. on-true コネクタを右クリックして、コンテキストメニューから**全てコピー**（子要素）を選択します。

コネクタが全てコピーに変更されます。



6. **core** ライブラリの**string functions** 以下からcontains 関数を挿入します。
7. **name** から**値** への接続を行い、**SearchFor** パラメーターから**substring** へ接続した後、result をフィルターのbool アイテムへ接続します。



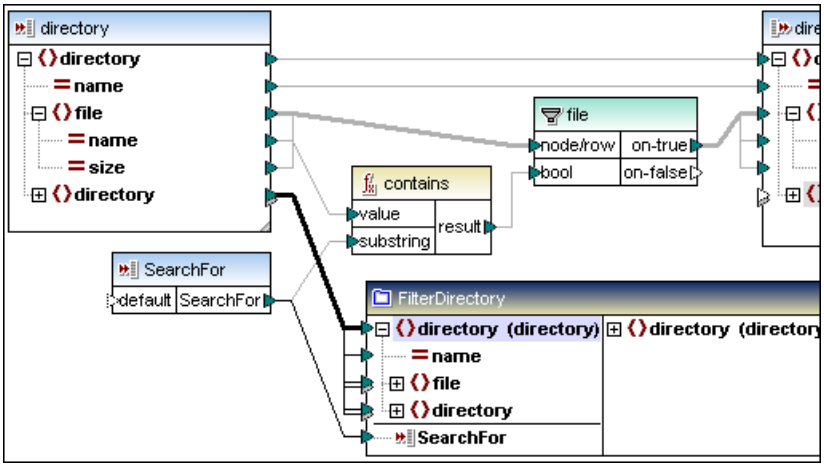
8. 入力コンポーネントのSearchFor アイテムから、ユーザー定義関数のSearchFor アイテムへ接続します。

再帰の定義を行う

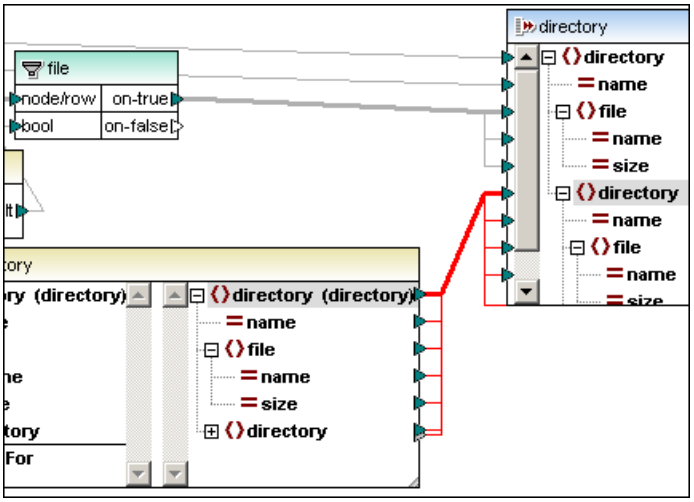
この段階で、与えられた 1 つのディレクトリ階層に対するマッピングが作成されました。次にサブディレクトリに対する処理を定義します。

アイコンバーの自動接続アイコン が有効になっていることを確認してください。

1. 入力コンポーネント下部にあるdirectory アイテムを、再帰的なユーザー定義関数の上部にあるdirectory アイテムへ接続します。



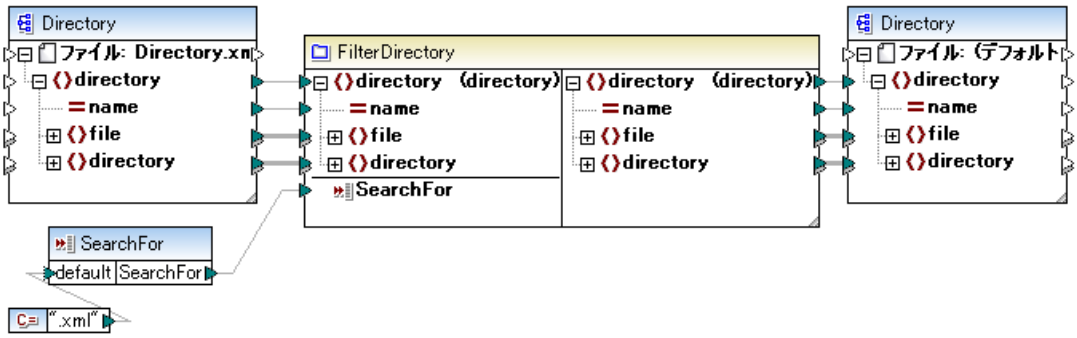
2. ユーザー定義関数の上部にあるdirectory アイテムを、出力コンポーネント以下にあるdirectory アイテムへ接続します。
3. 全てコピー接続を作成するか尋ねられた場合は、コネクタを右クリックして、コンテキストメニューから全てコピーを選択しをクリックします。



これでユーザー定義関数の定義を完了することができます。
メインマッピングウィンドウへ戻るアイコン をクリックして、マッピングの定義を継続します。

メインマッピングウィンドウ

1. ライブラリウィンドウの**user** セクションからFilterDirectory 関数を**メイン**マッピングエリアへドラッグします。
2. メニューオプションの 挿入 | XML スキーマファイル」からDirectory.xsd を挿入し、インスタンスファイルとして Directory.xml を選択します。
3. 同様の方法でDirectory.xsd を選択し、ルート要素の選択をスキップして出力コンポーネントを挿入します。
4. 定数コンポーネントを挿入し、SearchFor コンポーネントという名前の入力コンポーネントを挿入します。
5. 以下のスクリーンショットに有るような接続を作成します。
6. FilterDirectory ユーザー定義関数の入力ならびに出力側に接続されているdirectory アイテムのコネクタを右クリックして、コンテキストメニューから全てコピーを選択します。



7. 出力ボタンをクリックして、マッピングの結果を確認します。

メモ:
Directoryコンポーネントの最下部にある "directory" アイテムをダブルクリックすることで、1つ下のレベルにある再帰構造 (directory | file | directory) を確認することができます。全てコピー接続を使用することで、XML インスタンスファイル内にある全ての再帰レベルが使用されるようになり、手動で再帰レベルを展開する必要がなくなります

8.3 カスタム XSLT 1.0 または 2.0関数のインポート

XSLT 1.0 と2.0関数ライブラリをMapForce 内で使用することのできる単純型を返すカスタム関数として拡張することができます。

簡単なデータ型 (例えば、文字列) を返すカスタム関数のみがサポートされます。

XSLT ファイルから関数をインポートする:

1. **ツール** メニューから、**オプション**をクリックします。(または、ライブラリウィンドウの下のエリア内の**ライブラリの追加 削除**をクリックします。)
2. **ライブラリ**の横の**追加**をクリックし、.xsl または .xslt ファイルを参照します。

インポートされたXSLT ファイルは、ライブラリウィンドウ内のライブラリに表示され、全ての名前を持つテンプレートを関数として、ライブラリ名の下に表示されます。インポートされたライブラリが表示されない場合、XSLT が変換言語として設定されているかを確認してください (次を参照 :変換言語の選択)。

以下の点に注意してください::

- MapForce にインポートするには、関数は、XSLT ファイル内のXSLT 仕様に準拠する名前を持つテンプレートとして宣言されている必要があります。XSLT 2.0 ドキュメント内で発生する関数を`<xsl:function name="MyFunction">` の書式でインポートすることもできます。インポートされたXSLT ファイル内で、他のXSLT ファイルをインポートまたは含むことができ、これらのXSLT ファイルと関数もインポートされることができます。
- インポートされたカスタム関数のマップすることのできる入力コネクタはテンプレート呼び出し内で使用されるパラメーターの数により異なります。任意のパラメーターもサポートされています。
- 名前空間はサポートされています。
- MapForce に既にインポートされているXSLT ファイルの更新を行う場合、変更は自動的に検知され、MapForce がファイルの再ロードを促します。
- 名前付きのテンプレートに書き込む場合、テンプレート内で使用されているXPath ステートメントが正しい名前空間を持つことを確認してください。マッピングの名前空間にバインドする名前空間を確認するには、生成されたXSLT コードをプレビューしてください。.

XPath 2.0 内のデータ型

XML ドキュメントからXML スキーマが参照されており、スキーマに対してドキュメントが妥当である場合、演算にて目的のデータ型へ暗示的に変換されないデータ型を明示的に構築またはキャストする必要があります。

Altova XSLT 2.0 エンジンで使用されるXPath 2.0 データモデルでは、XML ドキュメントから**原子化**された全てのノード値にxs:untypedAtomic データ型が割り当てられます。xs:untypedAtomic 型は、以下にあるように暗示的な型変換に使用されます:

例えば、

- xs:untypedAtomic("1") +1 では、xdt:untypedAtomic 値が加算演算子によりxs:double に**暗黙的に**変換され、式の結果は 2 となります。.
- 四則演算では、オペランドが暗黙的にxs:double へ変換されます。
- 値の比較を行う場合、比較の前にオペランドがxs:string へ変換されます。

以下も参照してください:

例 :カスタムXSLT 1.0 関数の追加

XSLT 1.0 エンジンの実装
XSLT 2.0 エンジンの実装

8.3.1 例 :カスタム XSLT 1.0 関数の追加

この例は、カスタムXSLT 1.0 関数をMapForce にインポートする方法を説明しています。以下に示される単純なサンプルに必要なファイルは **...\ MapForceExamples** ディレクトリに収められています。

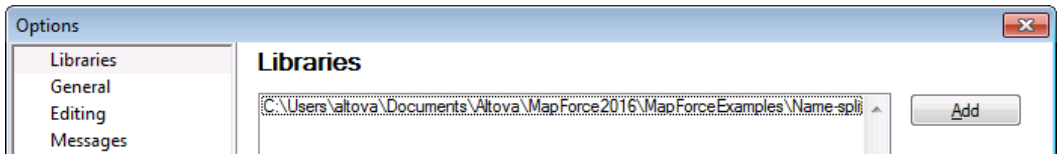
- 目的の変換を行い、その結果を出力するXSLT ファイルを作成します。
  以下に示される**Name-splitter.xslt** サンプルでは、"string" のパラメーターを伴う "**tokenize**" という名前付きテンプレートを確認することができます。テンプレートにより、空白スペースの後に来る文字が大文字になります。

```
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($string, '-abcdefghijklmnopqrstuv
16     <xsl:variable name="capscount" select="string-length($caps)"/>
17     <xsl:variable name="token">
```

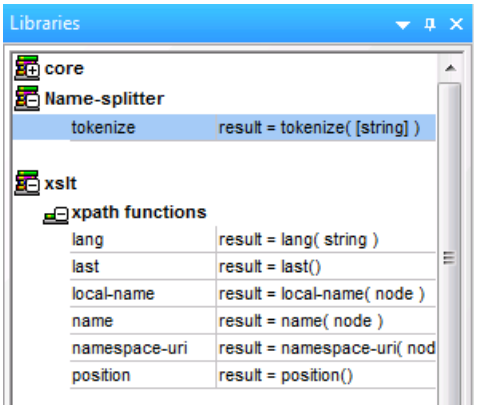
- **Name-splitter.xml** (処理されるソースXML インスタンスファイル)
- **Customers.xsd** (ソースXML スキーマ)
- **CompletePO.xsd** (ターゲットXML スキーマ)

カスタムXSLT 関数の追加 :

1. XSLT を変換言語として選択します。 (次を参照してください:変換言語の選択)
2. ライブラリウィンドウの下のエリア内の**ライブラリを追加 /削除** ボタンをクリックします。または、**ツール**メニューから**オプション**をクリックして、**ライブラリ**を選択します。
3. **追加** をクリックして、関数として使用する名前付きテンプレートを含むXSL またはXSLT ファイル (この例では **Name- splitter.xslt**) を選択します。

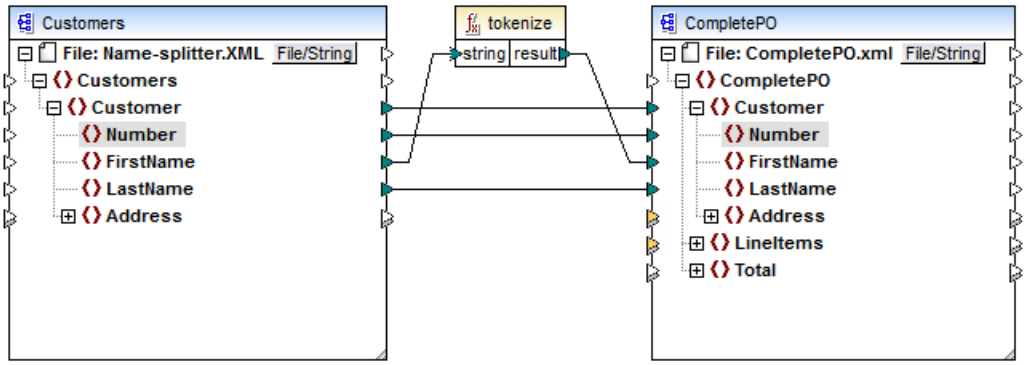


4. 「**OK**」をクリックします。名前付きテンプレートとして定義された関数とともにXSLT ファイル名がライブラリウィンドウに表示されます。この例では、**Name- splitter** というライブラリ以下に`tokenize` という関数が表示されます。



マッピング内でXSLT 関数を使用する:

1. マッピングウィンドウへ`tokenize` 関数をドラッグして、以下のスクリーンショットに表示されているようなマッピングを作成します。



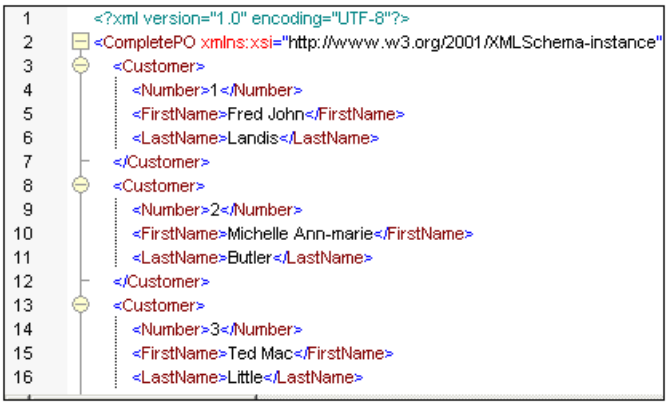
2. XSLT タブをクリックして、XSLT コードを生成します。




```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xsi="http://v
  <xsl:output method="xml" encoding="UTF-8"/>
  <xsl:include href="C:\Program Files\Altova\MAPFORCE2004\MapForceExamples\Name-splitter.xslt"/>
  <xsl:template match="/Customers">
    <CompletePO>
      <xsl:attribute name="xsi:noNamespaceSchemaLocation">C:/PROGRA~1/Altova/MAPFORCE200
      <xsl:for-each select="Customer">
        <Customer>
          <xsl:for-each select="Number">
            <Number>
              <xsl:value-of select="."/>
            </Number>
          </xsl:for-each>
          <xsl:for-each select="FirstName">
            <xsl:variable name="V47993824_47988944" select="."/>
            <xsl:variable name="V47993824_47939520">
              <xsl:call-template name="tokenize">
                <xsl:with-param name="string" select="$V47993824_47988944"/>
              </xsl:call-template>
            </xsl:variable>
```

メモ: 名前付きテンプレートがマッピングで使用されると、名前付きテンプレートが含まれているXSLT ファイルが、生成されたXSLT コードに **xsl:include href...** により**インクルード**され、**xsl:call- template** コマンドを使用することで呼び出されます。

3. 出力タブをクリックすることで、マッピングの結果を確認できます。



XSLT ライブラリを MapForce から削除する:

1. **ライブラリを追加/削除**ボタンをクリックします。
2. ライブラリタブにて、目的のXSLT **ライブラリ名**をクリックし、削除ボタンをクリックした後

8.4 カスタム XQuery 1.0 関数をインポートする

XQuery がマッピングの変換言語として選択されている場合、MapForce は、XQuery のために使用することのできる関数ライブラリをライブラリウィンドウ内に表示します。必要であれば、このリストをXQuery 関数を使用して、カスタム XQuery 1.0 ライブラリモジュールを MapForce にインポートして拡張することができます

MapForce にインポートするためには、XQuery ファイルは以下の条件を満たす必要があります :

- XQuery 仕様に従う有効なライブラリモジュールである必要があります。すなわち、`module namespace <prefix>="<namespace name"` などのモジュール宣言と共に開始する必要があります。i
- インポートされたライブラリモジュール内で宣言された全ての関数は、動的なデータ型を返す必要があります (例えば、`xs:string`, `xs:boolean`, `xs:integer` など)。関数パラメーターも動的な型を有する必要があります。

XQuery ライブラリモジュールをインポートする :

1. 「**ツール**」メニューから「**オプション**」をクリックします (または、ライブラリウィンドウの下のエリアの「**ライブラリの追加 / 削除**」をクリックします)
2. 「**ライブラリ**」の横から、「**追加**」をクリックして、xq または xquery ライブラリファイルを参照します。

インポートされた ライブラリモジュールがサポートされていない場合、メッセージボックスがプロンプトします。それ以外の場合、インポートされた ライブラリモジュールは、ライブラリウィンドウに表示されます。特定の関数をマッピングエリアにドラッグして、他の MapForce 関数コンポーネントと同様に使用します。

インポートされたXQuery がライブラリモジュールに表示されない場合、XQuery が変換言語として選択されていることを確認してください (次を参照してください 変換言語の選択)。

以下も参照 :

XQuery エンジンの実装

8.5 カスタム Java と .NET ライブラリのインポート

コンパイルされたJava クラスファイルや .NET アセンブリファイル (.NET 4.0 アセンブリを含む) はライブラリペインに追加して、MapForce にあるその他の関数と同じように使用することができます。これらJava や .NET 関数の出力は、出力ペインで確認することができるほか、関数は生成されたコードでも利用することができます。

メモ:

- カスタムJava または .NET 関数をインポートするには、 Java クラス (class) または .NET.dll アセンブリファイルをコンパイルする必要があります。.NET アセンブリではない、 Java .jar ファイルまたは dll ファイルのインポートはサポートされていません。
- コンパイルされたJava クラス (class) ファイルは、マッピング言語がJava に設定されているサポートされています。Java Runtime Environment 7 または以降が使用するマシンにインストールされている必要があります。特定の型とメンバーがサポートされています (以下を参照してください:Java 関数サポート)。
- .NET アセンブリファイルは、マッピング言語がC# に設定されているとサポートされています。パラメーターと戻り値の型としてシステムアセンブリから基本データ型 .NET アセンブリは、C# 以外の .NET 言語で書かれている可能性があります。(例えば、C++.NET またはVB.NET)。のみを使用することを考慮すると、 (以下を参照してください: .NET 関数サポート)。
- マッピングがインポートされたJava .class または .NET DLL アセンブリを使用する場合、マッピング言語をC++ に設定することは、サポートされていません。
- ネイティブのC++ DLL 関数をからインポートすることは、制限されており、特別なアプローチが必要です。詳細に関しては、以下を参照してください Java、C# とC++ ライブラリに手動で参照する
- マッピングがインポートされたJava .class または .NET DLL アセンブリを使用する場合、マッピング言語を XSLT に設定することは、サポートされていません。 (アダプターの役割をするカスタムXSLT 関数が書き込まれる必要があります)。
- MapForce マッピングから呼び出される全ての関数が "べき等" である必要があります (これは、同じ入力パラメーターを使用して関数が呼び出される都度、同じ値が返されることを意味します)。 MapForce により呼び出される正確な順序および回数は定義されていません。
- Java の場合、インポートされたクラスファイルとパッケージは、CLASSPATH 変数に追加される必要はありません。これは、ビルトイン実行エンジンが、生成されたJava コードと同様、Java エンジンのクラスパスまたはAnt の自動的にインポートされたパッケージに追加されるからです。しかしながら、インポートされたクラスファイルとパッケージの依存関係は、自動的には処理されません。ですから、インポートされたJava クラスファイルまたはパッケージが他のクラスファイルに依存する場合、CLASSPATH 環境変数の全ての依存パッケージの親ディレクトリが追加されていることを確認してください。

Java 関数サポート

トップレベルのクラス、静的メンバークラス、そして非静的メンバークラスがサポートされます:

- `new <classname>(<arg1>, <arg2>, ...)`
- `<object>.new <member-class>(<arg1>, <arg2>, ...)`

メンバー関数と静的関数がサポートされます:

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

XML スキーマとJava のデータ型間におけるサポートされている変換を以下に示します:

スキーマ型	Java 型
xs:string	String

スキーマ型	Java 型
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

両方向への接続を行うことができます。その他の Java データ型はサポートされていません。配列型はサポートされていません。配列を使った引数や戻り値は無視されます。.

Object 型はコンストラクター呼び出しまたはメソッドの戻り値としてサポートされています。これらの型はその他の Java メソッドへマッピングすることもできます。MapForce にてオブジェクトを操作することはできません。

.NET 関数サポート

トップレベルのクラス、静的メンバークラス、そして非静的メンバークラスがサポートされます：

- `new <classname>(<arg1>, <arg2>, ...)`

メンバー関数と静的関数がサポートされます：

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

以下に示される XML スキーマ型と .NET/C# 型との間の接続がサポートされます：

スキーマ型	.NET type	C# type
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

スキーマ型	.NET type	C# type
xs:double	System.Double	double
xs:decimal	System.Decimal	decimal

両方向への接続を行うことができます。その他の .NET/C# 型はサポートされていません。配列型はサポートされていません。配列を使った引数や戻り値は無視されます。

Object 型はコンストラクター呼び出しまたはメソッドの戻り値としてサポートされています。これらの型はその他の .NET メソッドへマッピングすることもできます。MapForce にてオブジェクトを操作することはできません。

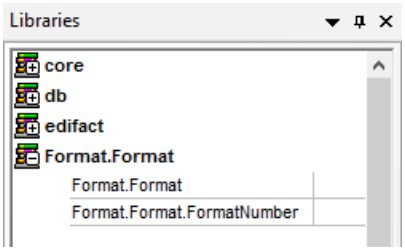
8.5.1 例 :カスタム Java クラスのインポート

この例は、カスタム Java クラスファイルを MapForce にインポートする方法を説明しています。

メモ: Java SE 7 Runtime Environment または以降が必須です。

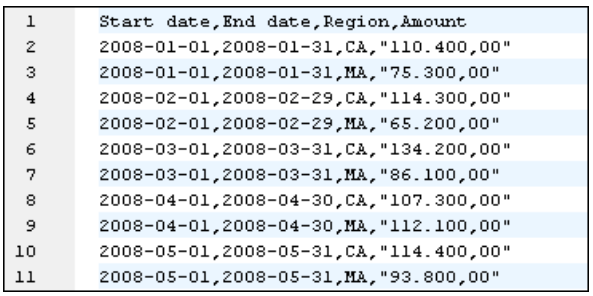
MapForce ライブラリとして Java .class ファイルを追加する:

1. 変換言語を Java に設定します (次を参照してください:変換言語の選択)。
2. ライブラリウィンドウの下のエリアの **「ライブラリの追加 削除」**ボタンをクリックします。
3. ライブラリの横から、 **追加」**をクリックして、**Format.class** ファイルを**\MapForceExamples\Java \Format** ディレクトリから選択します。新規のライブラリが追加されたことを通知するメッセージが表示されます。インポートされたライブラリはライブラリウィンドウで確認することができます。



MapForce 内でマッピングの出力をプレビューする方法 :

1. **...\MapForceExamples\Java** フォルダー内にある**FormatNumber.mfd** ファイルを開きます。
2. 出力ボタンをクリックして、マッピングの結果を確認します。



```
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"
```

Java 内でマッピングを実行する:

1. **「ファイル」**メニューから**Generate Code In | Java**をクリックします .
2. コードが生成されるターゲットディレクトリを選択し、「OK」をクリックします。

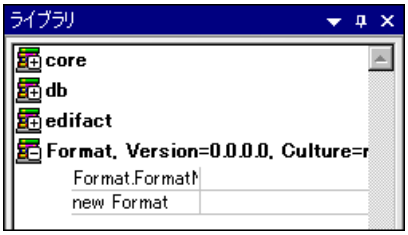
3. Java プロジェクトに生成されたライブラリをインポートし、Java アプリケーションを構築します (例えば、次を参照してください:例 :Eclipse とAnt を使用して Java アプリケーションを構築する)。

8.5.2 例 :カスタム .NET DLL アセンブリのインポート

この例は、C# で作成されたカスタムアセンブリを MapForce にインポートする方法を説明しています。

.NET アセンブリファイルを追加する:

1. 変換言語を C# に設定します (次を参照してください:変換言語の選択)。
2. ライブラリウィンドウの下のエリアの **「ライブラリの追加 削除」** ボタンをクリックします。
3. ライブラリの横から、click **「追加」** をクリックして、**Format.dll** ファイルを **\MapForceExamples\C#\Format\bin\Debug** ディレクトリから選択します。新規のライブラリが追加されたことを通知するメッセージが表示されます。インポートされたライブラリはライブラリウィンドウで確認することができます。



マッピングの出力をプレビューする:

1. **...\MapForceExamples\C#** フォルダー内にある **FormatNumber.mfd** ファイルを開きます。
2. 出力ボタンをクリックして、マッピングの結果を確認します。

```
 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"
```

マッピングをカスタム C# アプリケーションから実行する:

1. **「ファイル」** メニューから **「次でコードを生成 | C#」** をクリックします .
2. コードが生成されるターゲットディレクトリを選択し、**「OK」** をクリックします。
3. Visual Studio を使用してアプリケーションを作成し、生成されたコンソールアプリケーションを実行します (次を参照してください: C# コードの生成)。

8.6 Java、C# および C++ ライブラリを手動で参照する

MapForce に直接カスタムライブラリをインポートする代替のオプションとしては、MapForce により識別されているカスタム .mff ファイル (MapForce 関数ファイル) を使用してレファレンスを作成することができます。 mff ライブラリファイルは、カスタムコード内のクラス定義と MapForce のリンクを手動で定義する XML ファイルにより必須です。カスタム mff ファイルを作成すると、.NET DLL または Java クラスファイルをインポートすることと同様 MapForce にインポートすることができます。

メモ:

- インポートされた関数がライブラリウィンドウで表示されるようにするには、戻り値の型とパラメーターは、単純型である必要があります。各言語のために使用することのできるデータ型のリストに関しては、以下を参照してください:データ型マッピング。
- 関数ライブラリをカスタム mff ファイルからインポートすると、(**出力**) ボタンをクリックして) MapForce 内で直接マッピングをプレビューすることは制限されます。 C++ で書かれているライブラリに関しては、MapForce 内のマッピングのプレビューは、サポートされていません。Java と C# の場合は、ライブラリがネイティブ言語の型を使用する場合は、プレビューを使用することができますが、ライブラリが Altova 生成されたクラスをインポートする場合は、使用することができません。しかしながら、ライブラリによりターゲットされている特定の言語でコードを生成することができます。カスタム関数は、生成されたコード内で使用することができ、生成されたコードからマッピングを実行することが可能になります。
- 生成されたマッピングコードにより呼び出される関数の順序は定義されていません。MapForce は、再利用または式の評価のために計算された結果をキャッシュする可能性があります。このため、副作用を与えない、カスタム関数を使用することが奨励されます。
- ユーザー定義関数とカスタム関数ライブラリを区別する必要があります。ユーザー定義関数は、マッピング内で視覚的に作成され、マッピング mfd ファイルと共に保存されるため *mff ファイルには保存することができません。詳細に関しては、以下を参照してください:ユーザー定義関数の再利用 (インポート)。
- 2010 より以前の MapForce バージョンをアップグレードする場合、カスタム関数内で使用されているデータ型を更新する必要がある可能性があります (以下を参照してください:データ型マッピング)。

カスタム mff ファイルの作成と構成の順序に関する詳細は、以下を参照してください:.mff ファイルの構成。例に関しては、以下を参照してください:

- 例:カスタム C# ライブラリの作成
- 例:カスタム C++ ライブラリの作成
- 例:カスタム Java ライブラリの作成

8.6.1 .mff ファイルの構成

MapForce 関数ファイル (mff) は、MapForce にカスタム Java、C#、または C++ ライブラリから関数の適用を可能にし、ライブラリウィンドウ内で表示される XML フォーマットの構成ファイルです。 mff ファイルは、基本的にカスタムライブラリと MapForce の中間であり、a) カスタム関数にインターフェイスし、また、b) 生成されたコード内に実装が存在する箇所を指定するために構成されていなければなりません。

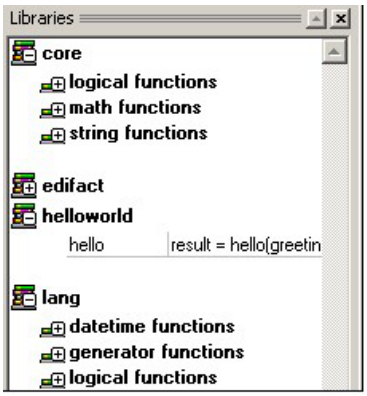
メモ: ***.mff** ライブラリファイルは、**Altova\MapForce2017\MapForceExamples**.. フォルダー内に収められている **mff.xsd** スキーマファイルに対して妥当である必要があります。スキーマは内部的な使用のためだけのもので、カスタムライブラリの構成が定義されます。Altova GmbH は、新規リリースにて、このファイルフォーマットを変更する権利を有しています。

C# に対応した基本的な mff ファイルの例を以下に示します:

```
<?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>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting sentence according to the given
greeting_type.</long>
         </description>
      </component>
   </group>
</mapping>
```

mff ファイルがMapForce で表示される例を以下のイメージに示します。新たなライブラリ "helloworld" が、hello 文字列関数を含む (アルファベット順に並べ替えられた) ライブラリエントリーに表示されます。



ニーズに合うかたちで mff ファイルを適応させるステップを以下に示します：

ライブラリ名の構成方法

以下に示されるmff ファイルのセグメントにてライブラリ名が宣言されます。慣例として、ライブラリ名は全て小文字で記述されます。

```
<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">
```

上のサンプルでは、ライブラリウィンドウに表示されるエントリは、"helloworld" と呼ばれます。

言語の実装の構成

<implementations> 要素はライブラリがサポートする言語を指定する必須要素であり、<mapping> の子として追加される必要があります。例：

```
...
<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>
...
```

各 <implementation> の設定により、生成されたコードからJava、C++、C# にて定義された関数を呼び出すことができるようになります。

.mff ファイルは、1つ以上の言語をターゲットすることができるように作成することができます。 この場合、各追加言語ごとに追加 <implementation> 要素が含まれる必要があります。各プログラミング言語の設定を以下に示します。

Java

```
...
<implementation language="java">
   <setting name="package" value="com.hello.functions"/>
   <setting name="class" value="Hello"/>
</implementation>
...
```

生成されたコードから**Hello.class** ファイルを発見できるようにしておく必要があります。ファイルまでのパスをJava CLASSPATH へ入力してください。デフォルトのクラスパスはシステム環境変数にて確認することができます。

カスタムJava ライブラリを利用する場合、*.mff ファイルごとに 1 つのクラスを割り当てられることに注意してください。

C#

```
...
   <implementation language="cs">
      <setting name="namespace" value="HelloWorldLibrary"/>
      <setting name="class" value="Hello"/>
      <setting name="reference" value=" C:\HelloWorldLibrary
\HelloWorldLibrary.dll"/>
   </implementation>
```

```
...
```

C# に関しては、コード内の名前空間が、.mff ファイル内で定義されている名前空間に一致することが重要です。(上のコードリストでの名前空間は、HelloWorldLIbrary です。) クラス名に関しても同様です (上のコードリストでの名前は Hello です。) 第3の設定である、reference は、生成されたコードへリンクするdll のパスを提供します。

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>
...
```

上のC++ サンプルリスティングに関して、以下に注意してください:

- namespace は、Greetings クラスが定義される名前空間です。この名前はライブラリ名と一致しなければなりません。
- path は、ソールファイルを見つけることができるパスです。
- マッピングのためのコードが生成されると、ソースファイルは (**「ファイル| コードを生成」**メニューオプションおよびディレクトリを選択する際に定義される) ディレクトリ**targetdir/libraryname** にコピーされ、プロジェクトファイルに含まれます

提供するファイルはすべて生成されるアルゴリズムに含まれます。

コンポーネントの追加

MapForce のライブラリウィンドウのグラフィカルユーザーインターフェイスでは、各関数は、関数グループの下にネストされ表示されます。例えば、「文字列関数」など。.mff ファイル内では、関数は<component> 要素に対応します。一方、各 <component> は、<group> 要素の下でネストされる必要があります。例:

```
...
<group name="string functions">
   <component name="hello">
      …
   </component>
</group>
...
```

下に表示されるコードは、hello と呼ばれるサンプル関数 (コンポーネント) を定義しています。

```
...
<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>
...
```

MapForce 内でコンポーネントは以下のように表示されます：



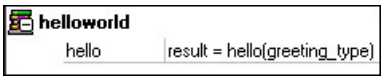
上のコードリスティングでは、<datapoint> は、大まかに(入力または出力コネクタとして知られる)関数の入力または出力パラメーターとして定義することができます。<datapoint> の type 引数は、パラメーターのデータ型 (または、戻り値のデータ型)を指定します。

各関数のために、ターゲットデータポイントのみが許可されています。定義することのできるソースデータポイントの数に制限はありません。

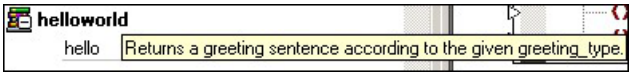
各データポイントのデータ型は、XML スキーマ型 (例えば、xs:string、xs:integer など)の1つである必要があります。これらのデータ型は、Java、C++ または C# ライブラリ内に定義された関数のパラメーターの型のデータ型に対応します。XML スキーマデータ型を言語型にマッピングするには、次を参照してください：データ型のマッピング。

ライブラリウィンドウ内では関数は短い、および、長い説明と共に表示されています。短い説明は常に、関数名の右側に表示されますが、長い説明は、マウスを短い説明の上にポイントするとヒントとして表示されます。

短い説明：



長い説明：



言語の実装の定義

ライブラリウィンドウ内で カスタム Java、C# または C++ クラスの関数間の接続を作成します。これは、<implementation> 要素を使用することにより達成することができます。

上記の通り、1つの関数が、サポートされるプログラミング言語のために1つづつ、複数の実装要素を持つ場合があります。関数は、Java 内では、"helloFunction" と呼ばれ、C++ 内では "HelloFunctionResponse" と呼ばれる場合があります。このため、各プログラミング言語のために個別の関数名を指定する必要があります。3つのプログラミング言語のための関数は以下に類似する可能性があります：

```
...
<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>
...
```

関数名として提供される値は、Java、C# または C++ クラス内のメソッドの名前に正確に一致する必要があります。

8.6.2 .mff ファイルを MapForce にインポートする

カスタム .mff ファイルの作成後（次を参照してください：.mff ファイルの構成）以下の方法で MapForce にインポートすることができます：

1. **「ツール」** メニューから **「オプション」** をクリックします（または、ライブラリウィンドウの下のエリアの **「ライブラリの追加 / 削除」** をクリックします）。
2. ライブラリの横に存在する **「追加」** をクリックして、**「追加」** をクリックし、カスタム .mff ファイルを選択してください。

マッピング言語をカスタムライブラリによりターゲットにされた言語に設定すると、インポートされたライブラリがライブラリウィンドウ内で表示されます。

*.mff ファイルを **「プログラムファイル」**（または **「プログラムファイル (x86)」** folder）に相対する **..\Altova\MapForce2017\MapForceLibraries** フォルダーに保存すると、MapForce の起動時にライブラリは自動的にライブラリウィンドウにロードされます。対応するライブラリファイル (*.mff) を削除または追加することにより、ライブラリとその関数の表示の切り替えを行うことができます。

8.6.3 データ型のマッピング

次のテーブルは MapForce に Java、C#、および C++ ライブラリを適応する、カスタム .mff ファイルを作成する際に関数戻り値の型とパラメーターの型としてサポートされるデータ型をリストしています。テーブルは、ネイティブとネイティブでは内データ型をリストしています。Altova date などのネイティブでは内データ型をサポートする場合、カスタム Java と C# ライブラリは Altova ライブラリへのレファレンスを含む必要があります。C++ の場合、Altova ライブラリは常にインポートされる必要があります。Altova ライブラリの生成方法に関しての詳細は、次を参照してください：コードジェネレーター。

XML スキーマ型	Java 型	C# 型	C++ 型
anyAtomicType	String	string	string_type
anySimpleType	String	string	string_type
anyURI	String	string	string_type
base64Binary	byte[]	byte[]	altova::mapforce::blob
boolean	boolean	bool	bool
byte	int	int	int
date	com.altova.types.Date	Altova.Types.DateTime	altova::DateTime

XML スキーマ型	Java 型	C# 型	C++ 型
	Time		
dateTime	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
dayTimeDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
decimal	java.math.BigDecimal	decimal	double
double	double	double	double
duration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
ENTITIES	String	string	string_type
ENTITY	String	string	string_type
float	double	double	double
gDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonthDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYear	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYearMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
hexBinary	byte[]	byte[]	altova::mapforce::blob
ID	String	string	string_type
IDREF	String	string	string_type
IDREFS	String	string	string_type
int	int	int	int
integer	java.math.BigInteger	decimal	__int64
language	String	string	string_type
long	long	long	__int64
Name	String	string	string_type
NCName	String	string	string_type
negativeInteger	java.math.BigInteger	decimal	__int64
NMTOKEN	String	string	string_type
NMTOKENS	String	string	string_type
nonNegativeInteger	java.math.BigInteger	decimal	unsigned __int64
nonPositiveInteger	java.math.BigInteger	decimal	__int64
normalizedString	String	string	string_type
NOTATION	String	string	string_type

XML スキーマ型	Java 型	C# 型	C++ 型
positiveInteger	java.math.BigInteger	decimal	unsigned __int64
QName	javax.xml.namespace.QName	Altova.Types.QName	altova::QName
short	int	int	int
string	String	string	string_type
time	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
token	String	string	string_type
unsignedByte	long	ulong	unsigned __int64
unsignedInt	long	ulong	unsigned __int64
unsignedLong	java.math.BigInteger	ulong	unsigned __int64
unsignedShort	long	ulong	unsigned __int64
untypedAtomic	String	string	string_type
yearMonthDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration

8.6.4 例 :カスタム C# ライブラリの作成

このトピックでは MapForce ライブラリウィンドウに表示されるように、サンプル C# ライブラリの作成方法と、.mff ファイルの構成方法について説明されています。

1. Visual Studio 内に新規クラスライブラリを作成します。関数は、`public static` として定義されていることに注意してください。

```
using System;

namespace HelloWorldLibrary
{
    public class Greetings
    {
        public static string HelloFunction(bool GreetingType)
        {
            if (GreetingType)
                return "Hello World!";
            return "Hello User!";
        }
    }
}
```

2. 特別な XML スキーマ型 (日付および期間) が必要な場合は Visual Studio プロジェクトから **Altova.dll** ライブラリにレファレンスを追加する必要があります。このライブラリを取得するには、カスタム関数を使用しないマッピングから C# コードを生成します。**Altova.dll** ファイルはコードが生成されたディレクトリに相対する **..\Altova\bin\debug** ディレクトリ内にあります。Visual Studio 内の **Altova.dll** にレファレンスを追加するには、**「プロジェクト」** メニューから **「レファレンスを追加」** をクリックして、**Altova.dll** ファイルを参照します。また、コードに以下のラインを追加してください :`using Altova.Types;` 。 XML スキーマ型を C# 型にマップする方法に関する情報は、以下を参照してください :データ型のマッピング.
3. Visual Studio プロジェクトを構築します。**HelloWorldLibrary.dll** がプロジェクト出力ディレクトリに生成されます。

4. XML エディターを使用して、新規の mff ファイルを作成し、**..\Program Files\MapForceLibraries\mff.xsd** フォルダーに対して検証します。下でハイライトされているテキストが**HelloWorldLibrary.dll** ファイルを指していることを確認してください。 詳細に関しては、以下を参照してください：.mff ファイルの構成。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="HelloWorldLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="reference" value="C:\Projects
\HelloWorldLibrary.dll"/>
      </implementation>
   </implementations>
   <group name="Greetings">
      <component name="HelloFunction">
         <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>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

MapForce に適応するカスタムライブラリと mff ファイルの作成が完了しました。カスタム mff ファイルを MapForce で使用することができるようになりました(以下を参照してください：.mff ファイルを MapForce にインポートする)。

8.6.5 例 :カスタム C++ ライブラリの作成

このトピックでは MapForce ライブラリウィンドウに表示されるように、サンプル C++ ライブラリの作成方法と、mff ファイルの構成方法について説明されています。

1. ヘッダー (h) ファイルをクラスライブラリのために作成します。 以下のコードリストは**Greetings.h** という名前のサンプルヘッダーファイルを表しています。

```
   #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
```

関数は静的として宣言されており、名前空間 `altova` がインポートされています。
`ALTOVA_DECLSPECIFIER` をクラス名の前に書き込んでください。これにより、動的または静的なリンケージの使用に関わらず、次に生成されるコード内でクラスが正確にコンパイルされます。

2. .cpp ヘッダーファイルと同じ名前のファイルを作成します。 cpp ファイルは、 h ファイルと同じディレクトリに存在する必要があります。次のコードリスティングは、前に作成された**Greetings.h** ファイルを含む、**Greetings.cpp** と呼ばれるサンプル cpp ファイルを示しています：

```
#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!");
   }

}
```

StdAfx.h と一部の Altova ライブラリをインポートするラインに注意してください。これらのラインは、変更されてはなりません。ライブラリへのパスは正確です。生成されたコード内では、これらのパスはそれぞれのファイルを指します。

Java または C# と対象的に、ソース C++ ファイルをコンパイルする必要はありません。ファイルは生成されたコードにコピーされ、生成されたマッピングコードの残りと共にコンパイルされます。

3. XML エディターを使用して、新規 .mff ファイルを作成し、**..\Program Files\MapForceLibraries\mff.xsd** フォルダーに対して検証します。下でハイライトされているテキストが、前に作成されたヘッダーと cpp ファイルのディレクトリを指していることを確認してください。.mff ファイルの構成で説明されている通り、ここで定義されている名前空間と関数名、およびデータの型は、C++ コードに対応している必要があります。データ型へのサポートに関する詳細は、次を参照してください：データ型のマッピング。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
```

```
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="helloworld"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Projects\HelloWorld"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="Greetings">
      <component name="HelloFunctionResponse">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="HelloFunctionResponse"/>
            </implementation>
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

カスタムライブラリおよびMapForce を適応する .mff ファイルが作成されました。カスタム .mff ファイルをMapForce 内で使用することができます（次を参照してください：.mff ファイルをMapForce にインポートする）。ネイティブのC++ ライブラリを使用するマッピングを実行するには、C++ コードを生成し、C++ コードまたはアプリケーションから実行する必要があります。

RC++ コンパイルエラーが発生した場合

以下の行でコンパイラーエラーが発生した場合、msado15.DLL へのパスを追加してください。

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

Visual Studio 2008:

1. **「ツール | オプション」**を選択します。
2. 「プロジェクトおよびソリューション」の「VC++ ディレクトリ」タブを展開します。
3. ディレクトリを表示するプロジェクトのプルダウンから「インクルードファイル」を選択します。**msado15.dll** ファイルは以下で見つけることができます（通常 **C:\Program Files\Common Files \System\ADO**）。
4. 再度ビルドを行います。

8.6.6 例 :カスタム Java ライブラリの作成

このトピックでは MapForce ライブラリウィンドウに表示されるように、サンプル Java ライブラリの作成方法と、.mff ファイルの構成方法について説明されています。

1. (例えば、Eclipse などの)優先する開発環境で新規の Java プロジェクトを作成します。
2. `Hello` と呼ばれるクラスから構成される `com.hello.functions` と呼ばれる新規のパッケージをプロジェクトに追加します。下のコードリスティングでは、`HelloFunction` 関数は、`public static` として定義されていることに注意してください。

```
package com.hello.functions;

public class Hello {
   public static String HelloFunction ( boolean greetingType ) {
      if( greetingType )
         return "Hello World!";
      return "Hello User!";
   }
}
```

3. 任意で、日付、時間、または期間などの特別なスキーマ型をサポートする必要がある場合、`com.altova.types` パッケージをインポートしてください。このパッケージを取得するには、Java コードをカスタム関数を持たないマッピングから生成してください。

```
import com.altova.types.*;
```

4. クラスファイルへカスタムライブラリをコンパイルし、Java クラスパスに追加します。
5. XML エディターを使用して、新規の .mff ファイルを作成し、**..\Program Files\MapForceLibraries\mff.xsd** フォルダーに対して検証します。下でハイライトされているテキストがコード内で以前定義されている名前空間とクラスを指していることを確認してください。詳細に関しては、以下を参照してください:.mff ファイルの構成。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="java">
         <setting name="namespace" value="com.hello.functions"/>
         <setting name="class" value="Hello"/>
      </implementation>
   </implementations>
   <group name="Greetings">
      <component name="HelloFunction">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="java">
               <function name="HelloFunction"/>
            </implementation>
```

```
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

MapForce に適応するカスタムライブラリと .mff ファイルの作成が完了しました。カスタム .mff ファイルを MapForce で使用することができるようになりました (以下を参照してください: .mff ファイルを MapForce にインポートする)。

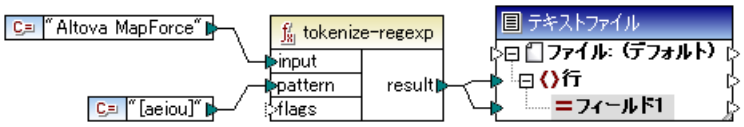
8.7 正規表現

MapForce では**match-pattern** と**tokenize-regexp** 関数 **pattern** パラメーターにて正規表現を使用して、入力パラメーターから得られた文字列の高度な絞り込みを行うことができます。

正規表現の構文ならびにセマンティクスは XSLT や XQuery で使用されているものと同一で、http://www.w3.org/TR/xmlschema-2/ にて定義されています。プログラミング言語によっては、正規表現の構文が若干違ってくることに注意してください。

用語

input	正規表現が適用される文字列
pattern	正規表現
flags	オプションのパラメーターで、正規表現の解釈方法を定義する
result	関数から得られる結果



tokenize-regexp 関数により文字列のシーケンスが得られます。行アイテムへの接続により、シーケンスのアイテムごとに新たな行が作成されます。

正規表現構文

リテラル (単一の文字)
例 :"a" という文字は最も基本的な正規表現となります。この正規表現により、文字列内にある最初の "a" という文字がマッチします。

文字クラス [ ]
角括弧に含まれる全ての文字セットがマッチの対象となります。

角括弧に含まれる文字の**1 つ だけ**が文字列に対してマッチします。

pattern **[aeiou]**
小文字の母音アルファベット全てがマッチの対象となります。

pattern **[mj]ust**
"must" または "just" という単語がマッチします。

"pattern" に含まれる値は大文字と小文字で区別されるという点に注意してください。小文字の "a" は大文字の "A" にマッチしません。

文字範囲 **[a-z]**
2 つの文字間にある文字全てがマッチの対象となります。一度にマッチする文字は一文字だけである点に注意してください。

pattern **[a-z]**
a から z まで、全ての小文字アルファベットがマッチの対象となります。

否定クラス **[^]**
角括弧の開始直後にハット記号を使用することで、その文字クラスを否定したことになります。

pattern **[^a-z]**
文字クラスに含まれていない (つまりこの例では小文字以外) 全ての文字にマッチします。

メタ文字 "."

ドットメタ文字
(改行を除く) **全ての文字**にマッチします。

pattern **.**
全ての単一文字にマッチします。

量指定子 ? + * {}

量指定子を使用することで、与えられた文字列にマッチする繰り返しを定義することができます。

?
0 または 1　　その前の文字列 (チャンク) がオプションとなります。

+
1回以上　　その前に表示される文字が1回以上マッチします。

*****
0回以上　　その前に表示される文字が0回以上マッチします。

{}
最小 (/最大) 繰り返し　　与えられた回数の繰り返しにマッチします。

例 :mo{1,3} により、"mo"、"moo"、"mooo" がマッチします。

()
サブパターン
正規表現の要素をグループ化するために使用されます。

|
代替 (/または) 左から順にサブパターンのマッチを行います。
例：**(horse|make) sense** により、"horse sense" または "make sense" がマッチします。

フラグ

以下に示されるのはオプションのパラメーターで、正規表現の解釈方法を指定するために使用されます。個々の文字によりオプションを指定することができます。文字はどの順序でも使用することができ、繰り返して使用することもできます。

s
このオプションが指定された場合、ドットの適用範囲が拡張されます。

メタ文字の "." が全ての文字にマッチするようになります。**s** フラグがセットされていれば、入力文字列に "hello" と "world" が 2 つの異なる行にある場合でも、"hello.*world" という正規表現によりマッチが行われます。

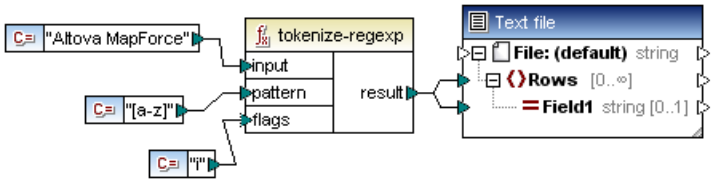
m
このオプションが指定された場合、複数行モードによるマッチが行われます。

複数行ノードでは、ハット記号 ^ が**全て**の行頭（文字列全体の開始位置と改行文字の直後に来る文字）に対してマッチするようになります。

更にドル記号 **$** が**全て**の行末（文字列全体の終了位置と改行文字の直前に来る文字）に対してマッチするようになります。

改行に使用される文字は #x0A となります

i
このオプションが指定された場合、マッチングにて大文字と小文字の違いが無視されます。
i フラグを指定すると、[a-z] によりa-z の文字とA-Z の文字がマッチするようになります。



x
このオプションが指定されている場合、マッチング処理が行われる前に正規表現文字列内にある空白スペースが削除されます。空白スペースの文字は #x09、#x0A、#x0D、そして #x20 になります。

例外：
文字クラス表現内の空白文字（例：[#x20]）は削除されません。

メモ：
コードを生成する場合、言語によっては正規表現構文の正確な機能や振る舞いが異なる可能性もあります。ご利用になる言語の正規表現がどのように動作するかを確認するようにしてください。

8.8 関数ライブラリレファレンス

このレファレンスチャプターでは、ライブラリ別に整理された『ライブラリ』ペイン内で使用することのできるMapForce 内蔵の関数に関して説明されています。

『ライブラリ』ペイン内の関数ライブラリは、選択された変換言語により異なります (変換言語の選択 を参照してください)。**core** ライブラリは、C++、C#、Java 言語で以下のうち1つで使用することのできる関数のコレクションです :XQuery、XPath、またはXSLT. **lang** ライブラリは、言 C++、C#、とJava 語内で使用することのできる関数です。他のライブラリは、出力の異なる型に関連した関数を含みます。

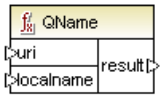
XPath 2.0 制約: シーケンスに関連するXPath 2.0 関数の一部は現在使用することができません。

8.8.1 core | QName functions

QName 関数は、XML ドキュメント内の修飾名 (QName) を操作する方法を提供します。

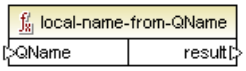
8.8.1.1 QName

名前空間 URI とローカル部分から構成されたQName が返されます。この関数を使用して、ターゲットコンポーネント内にQName を作成します。uri とローカル名パラメーターは定数関数により提供されます。

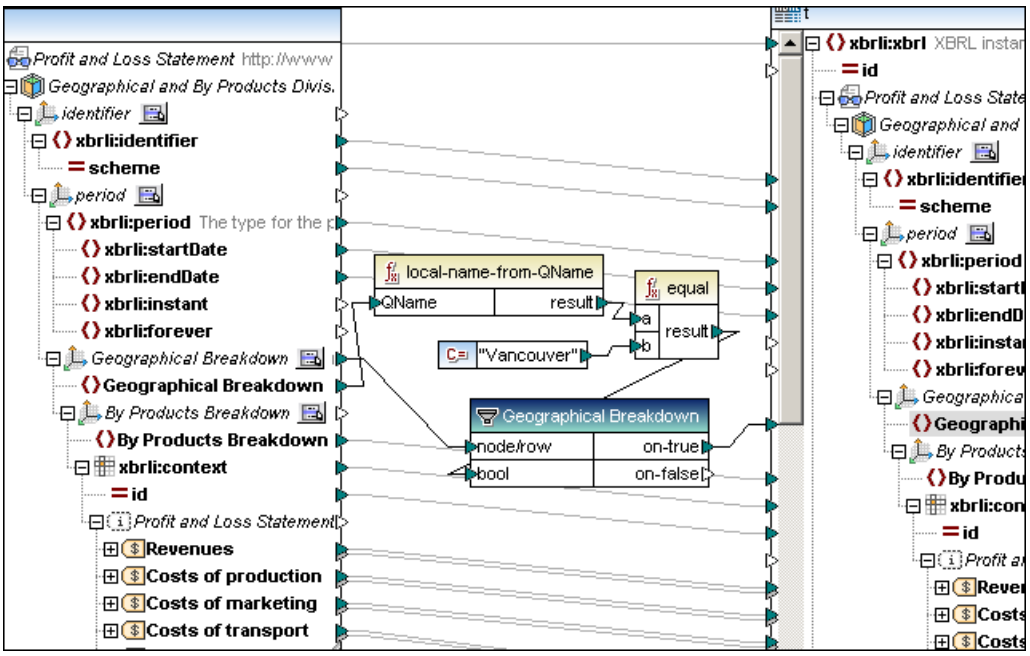


8.8.1.2 local-name-from-QName

QName のローカル名部分が返されます。



ハイパーキューブを含むXBRL インスタンスドキュメントをマップする際この関数は便利です。



マッピングは、明示的なメンバー (d-g:Vancouver) の**コンテンツ**の**ローカル名**が "Vancouver" に等しい箇所で、これらのファクトをフィルターします。メンバーのコンテンツ自身がQName であることに注意してください。

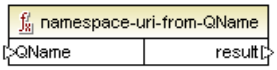
```
<xbrli:context id="Year2007_Vancouver_BeveragesTotal">
  <xbrli:entity>
    <xbrli:identifier scheme="http://www.stockexchange/ticker">ACME</xbrli:identifier>
    <xbrli:segment>
      <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-g:Vancouver</xbrl
      <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-b:BeveragesTotal</x
    </xbrli:segment>
```

ディメンションGeographicalbreakdown に属する全てのファクトはフィルターされ、ターゲットコンポーネントにパスされます。

```
<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:Beverages
      <explicitMember dimension="d-g:GeographicalBreakdown" xmlns="http://xbrl.org/2006/xbrldi" />
    </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:Cos
<p:CostsOfMarketing contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">1</p:Cost
<p:ProfitLoss contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">2</p:ProfitLoss>
```

8.8.1.3 *namespace-uri-from-QName*

QName の名前空間 URI 部分が返されます。



8.8.2 core | aggregate functions

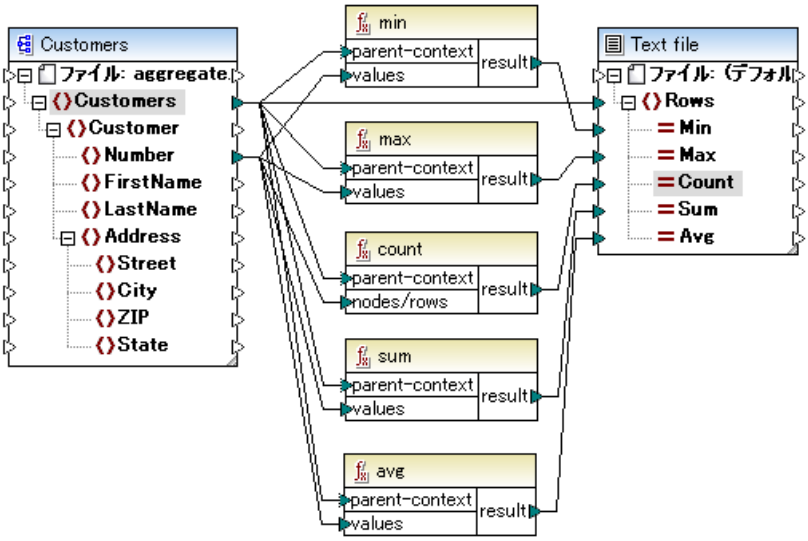
aggregate 関数は入力値のシーケンスやセットに対して使用されます。min、max、sum、そして avg の入力データが **decimal** データ型に変換され、処理されます。

- 入力値は関数の**values** パラメーターへ接続する必要があります。
- コンテキストノード（アイテム）を**parent-context** パラメーターへ接続して、入力シーケンスからデフォルトのコンテキストをオーバーライドすることができます。parent-context パラメーターはオプションになります！
- 関数の**result** を目的のターゲットアイテムへ接続します。

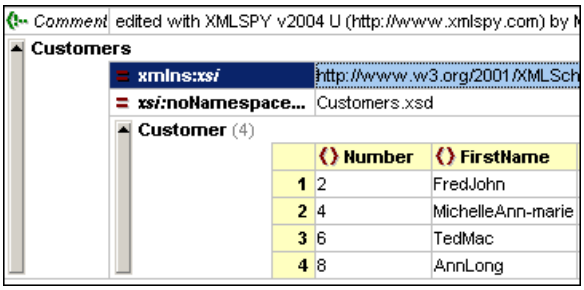
以下に示されるマッピングは .\Tutorial フォルダーに収められている**Aggregates.mfd** から確認することができます。

集計関数には 2 つの入力アイテムが備わっています。

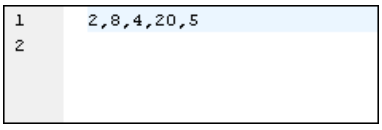
- **values** (node/row) にはデータを取得するためのソースアイテムが接続されます（この場合は Number）。
- **parent-context** にはコンテキストとなるアイテムが接続されます。このパラメーターはオプションです。



ここでの入力インスタンスは XML ファイルで、以下のデータが含まれています：



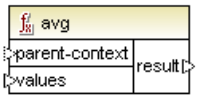
- ソースデータにある値は 2、4、6、8 という数値のシーケンスになります。
- ここでの出力コンポーネントはテキストファイルとなります。
  出力ボタンをクリックすることで以下の結果を得ることができます：



min=2、max=8、count=4、sum=20 とavg=5。

8.8.2.1 avg

入力シーケンスから得られた値の平均値を返します。入力が空セットの場合、結果は空セットになります。XSLT1 では利用できません。

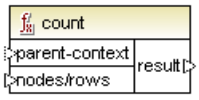


引数	説明
parent-context	オプションの引数。 親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
values	この引数は、実際のデータを提供するソースアイテムに接続されている必要があります。 与えられる引数の値は数値である必要があります。

使用方法の例に関しては、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリ内のマッピング **GroupTemperaturesByYear.mfd** を参照してください。

8.8.2.2 count

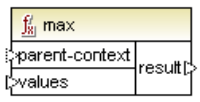
入力シーケンスから得られた値の数を返します。入力が空セットの場合、出力は 0になります。XSLT1 では機能が制限されます。



引数	説明
parent-context	オプションの引数。 親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
nodes/ rows	この引数は数えられるソースアイテムに接続されている必要があります。

8.8.2.3 max

入力シーケンスから得られた値の最大値を返します。入力が空セットの場合、結果は空セットになります。XSLT1 では利用できません。



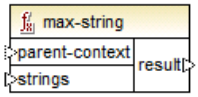
引数	説明
parent-context	オプションの引数。 親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。

引数	説明
values	この引数は、実際のデータを提供するソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。文字列のシーケンスから最大値を取得するためには、`max-string` 関数を使用します。

使用方法の例に関しては、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリ内のマッピング **GroupTemperaturesByYear.mfd** を参照してください。

8.8.2.4 max-string

入力シーケンス内の全ての文字列の値の最大の値を返します。例 :`max-string("a","b","c")` は、"c" を返します。この関数は、XSLT1 では利用できません。

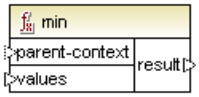


引数	説明
parent-context	オプションの引数。 親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
strings	この引数は、実際のデータを提供するソースアイテムに接続されている必要があります。与えられた引数の値は `xs:string` のシーケンス(ゼロまたは複数)である必要があります。

strings 引数が空のセットの場合、関数は空のセットを返すことに注意してください。

8.8.2.5 min

入力シーケンスから得られた値の最小値を返します。入力が空セットの場合、結果は空セットになります。XSLT1 では利用できません。

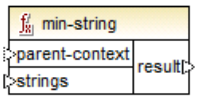


引数	説明
parent-context	オプションの引数。 親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
values	この引数は、実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。文字列のシーケンスから最小値を取得するためには、`min-string` 関数を使用します。

使用方法の例に関しては、**<Documents>\Altova\MapForce2017\MapForceExamples** ディレクトリ内のマッピング **GroupTemperaturesByYear.mfd** を参照してください。

8.8.2.6 min-string

入力シーケンス内の全ての文字列の値の最小の値を返します。例：`min-string("a", "b", "c")` は、"a" を返します。この関数は、XSLT1 では利用できません。

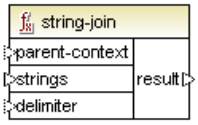


引数	説明
parent-context	オプションの引数。親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
strings	この引数は、実際のデータを与えるソースアイテムに接続されている必要があります。与えられた引数の値は `xs:string` のシーケンス(ゼロまたは複数)である必要があります。

文字列の引数が空のセットの場合、関数は空のセットを返すことに注意してください。

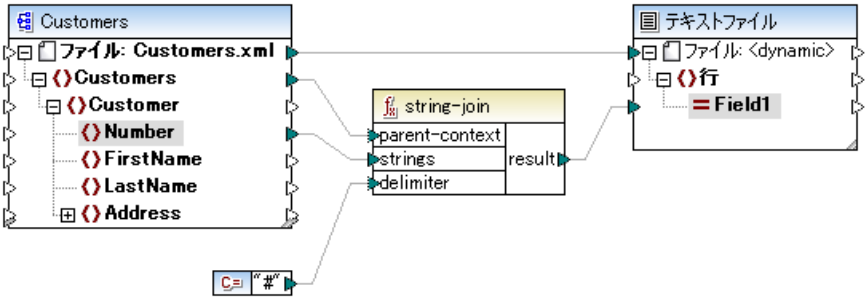
8.8.2.7 string-join

delimiter にて与えられた文字を区切り文字として、入力シーケンスから与えられた値を連結します。入力が空セットの場合、結果は空セットになります。XSLT1 では利用できません。



以下に示されるサンプルでは、Number に4つの値 (2、4、6、8)が与えられています。定数コンポーネントからは、ハッシュを表す "#"がdelimiter として与えられています。

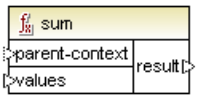
結果 = 2#4#6#8



delimiter (区切り記号)が与えられていない場合、デフォルトの区切り文字である空文字が使用され、区切り文字が表示されることはありません (上の例の場合、結果は 2468 となります)。

8.8.2.8 sum

入力シーケンスから得られた値の合計値を返します。入力セットが空の場合、結果は 0 になります。XSLT1 では利用できません。



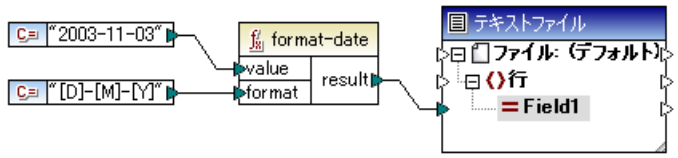
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。マッピングコンテキストのオーバーライド参照。
values	この引数は、実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。

以下を参照 :例 : ノードの値の加算。

8.8.3 core | conversion functions

明示的なデータ型変換をサポートするには、**conversion** ライブラリ内で複数の変換関数を使用することができます。多くの場合、MapForce は、必要な変換を自動的に作成し、これらの関数は特別な場合のみに使用される必要があります。

入力ノードが異なる型の場合、例えば、整数と文字列、変換関数を使用して文字列または数値の比較を行うことができます。



上のサンプルでは、最初の定数は型文字列で、文字列 "4"を含んでいます。2番目の定数は、定数 12 を含んでいます。2つの値を明示的に比較するには型が一致しなくてはなりません。

number 関数を最初の定数に追加するには、文字列の定数を数値 4 に変換します。比較の結果は "true" になります。

数値関数が使用されない場合、例えば、4 がパラメーターに直接接続されている場合、文字列の比較が生じ、結果は false になります。

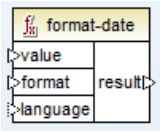
8.8.3.1 boolean

入力により得られた数値型の値をブール型へ変換します。入力が 0 の場合は "false" が返され、1 の場合は "true" になり、(equal や greater といった) logical 関数や、フィルター、または if-else 関数の入力として使用することができます。



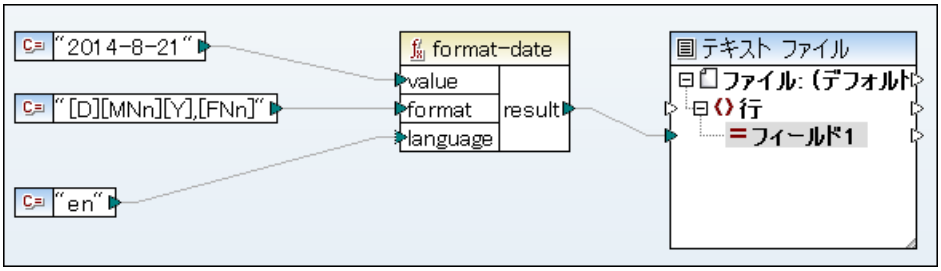
8.8.3.2 format-date

xs:date 入力の値を、特定のオプションに従い文字列にフォーマットし変換します。



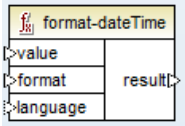
引数	説明
value	フォーマットされる日付。
format	日付がフォーマットされる方法を識別する文字列をフォーマットします。この引数は、**format-dateTime** 関数内の format 引数 と同じ方法で使用されます。
language	オプションの引数。与えられると、つきの名前と週の曜日が特定の言語で返されます。有効な値： **en (デフォルト)** 英語 **es** スペイン語 **de** ドイツ語 **ja** 日本語

下のサンプルでは、出力の結果は、以下の通りです："2014年、8月、14日、木曜日 "。この値をスペイン語に翻訳するには、引数の値をes に設定します。



8.8.3.3 format-dateTime

日時の値 (xs:dateTime) を文字列に変換します。日時の文字列の表示はformat 引数の値に従いフォーマットされます。



引数	説明
value	変換する xs:dateTime 型の値です。

引数	説明
format	**値**がフォーマットされる方法を識別するフォーマット文字列。
language	オプションの引数。与えられると、つきの名前と週の曜日が特定の言語で返されます。有効な値： **en (デフォルト)** 英語 **es** スペイン語 **de** ドイツ語 **ja** 日本語

メモ：
値はターゲットの型へキャストされるため、関数の出力（result）がstring 型以外のノードへ接続されている場合、書式が失われる可能性があります。ターゲットコンポーネントのコンポーネント設定にて**値をターゲットの方にキャスト**チェックボックスのチェックを外すことで、自動キャストを無効にすることができます(次を参照してください：コンポーネント設定の変更)。

`format` 引数は、角括弧に囲まれた変数マーカーと呼ばれる文字列により構成されます。角括弧の外にある文字は、定数として結果に反映されます。角括弧を定数文字として使用する場合、括弧を重ねあわせて記述する必要があります。

各変数マーカーは、表示する日付や時刻を表す識別子、オプションの書式修飾子、プレゼンテーション修飾子、幅修飾子により構成され、これらオプションの修飾子はコンマにより分割されます。

```
format := (literal | argument)*
argument := [component(format)?(presentation)?(width)?]
width := , min-width ("-" max-width)?
```

コンポーネントは以下のとおりです：

識別子	説明	デフォルトの表示
Y	年（絶対値）	4 桁の数値（2010）
M	その年の月	1- 12
D	その月の日付	1- 31
d	その年の日付	1- 366
F	その週の曜日	曜日の名前（言語に依存）
W	その年の週	1- 53
w	その月の週	1- 5
H	時（24時間）	0- 23
h	時（12時間）	1- 12
P	A.M. または P.M.	アルファベット（言語に依存）
m	その時間の分	00- 59
s	その分の秒	00- 59
f	小数点以下の秒	数値、小数点 1つ
Z	UTC からのタイムゾーンオフセット	+08:00

識別子	説明	デフォルトの表示
z	GMT からのタイムゾーンオフセット	GMT+n

書式修飾子は以下であることができます：

文字	説明	使用例
1	最初に 0を伴わない数値書式：1,2,3,...	1, 2, 3
01	2 桁の数値書式：01、02、03、...	01, 02, 03
N	コンポーネント名（大文字）	MONDAY, TUESDAY [1)]
n	コンポーネント名（小文字）	monday, tuesday [1)]
Nn	コンポーネント（頭文字が大文字）	Monday, Tuesday [1)]

メモ： N、n、Nn 修飾子は以下のコンポーネントでのみサポートされます：M、d、D。

width 修飾子がある場合、コンマと一緒に表示され、以下のような形式になります：

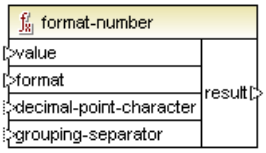
```
, min-width ("-" max-width)?
```

format-dateTime 関数を使用して、下のテーブルは xs:dateTime 値のフォーマットのサンプルを表示しています。"Value" 列は、**value** 引数に与えられる値を指定します。"Format" 列は、**format** 引数の値を指定します。"Result" 列は、関数により返される内容を表示しています。

日時	書式文字列	結果
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] [H]:[m]:[s].[f]`	2010 June 02 Wed 153 8:02:12.054
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f] [z]`	2010 June 02 Wed 153 8:02:12.054 GMT+02:00
2010-06-02T08:02	`[Y] [MNn] [D1] [F] [H]:[m]:[s].[f] [Z]`	2010 June 2 Wednesday 8:02:12.054 +02:00
2010-06-02T08:02	`[Y] [MNn] [D] [F,3-3] [H01]:[m]:[s]`	2010 June 2 Wed 08:02:12

8.8.3.4 format-number

XSLT 1.0 やXSLT 2.0、Java、C#、C++ や内蔵の実行エンジンで利用することができます。



引数	説明
value	必須の引数。フォーマットする値。
format	必須の引数。数値を整形（フォーマット）するためのフォーマット文字列。 この引数は**format-dateTime** 関数内の format 引数と同じ方法で使用されます。
decimal-point-format	オプションの引数。小数点に使用される文字。デフォルトでは "." になります
grouping-separator	オプションの引数。数値グループを分割するのに使用されるセパレーター /デリミタ。デフォルトでは "," になります。

メモ： 値はターゲットの型へキャストされるため、関数の出力（esult) がstring 型以外のノードへ接続されている場合、書式が失われる可能性があります。ターゲットコンポーネントのコンポーネント設定にて**値をターゲットの方にキャスト**チェックボックスのチェックを外すことで、自動キャストを無効にすることができます。

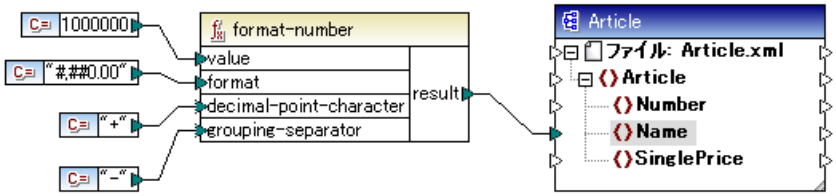
フォーマット：

```
format := subformat (;subformat)?
subformat := (prefix)? integer (.fraction)? (suffix)?
prefix := 特別文字以外の文字
suffix := 特別文字以外の文字
integer := (#)* (0)* ( allowing ',' to appear)
fraction := (0)* (#)* (allowing ',' to appear)
```

最初のsubformat が正の数のフォーマットに使用され、 2番目のsubformat が負の数に対して使用されます。subformat が 1つだけ指定された場合、同じsubformat が正の数と負の数に使用され、負の数の場合はprefix の前にマイナス記号が挿入されます。

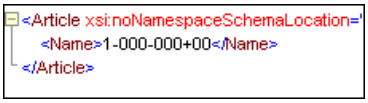
特別な文字	デフォルト	説明
zero-digit	0	result にてこの場所に常に数値が表示されます。
digit	#	最終的に 0になる場合を除き、result の文字列にてこの場所に常に数値が表示されます。
decimal-point	.	数値の整数部と小数点以下を分けるために使用されます。
grouping-seperator	,	数値のグループ分けを行うのに使用されます。
percent-sign	%	数値を100倍して、パーセンテージとして表示します。
per-mille	‰	数値を1000倍して、パーミルとして表示します。

decimal-point-character とgrouping-separator に使用される文字は、それぞれ常に "."と ","になります。しかしこれらのノードに定数をマッピングすることで、出力されるフォーマットを変更することができます。



上に示されるformat-number 関数のマッピング結果を以下に示します：

- decimal-point-character が "+"に変更されました。
- grouping-separator が "-"に変更されました。



四捨五入

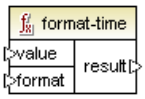
端数処理には四捨五入が使用されます。つまり端数が 0.5以上であれば切り上げが行われ、0.5未満であれば切り捨てが行われます。このメソッドは、生成されたコードとビルトイン実行エンジンでのみ使用することができます。

XSLT 1.0 では四捨五入のモードが定義されておらず、XSLT 2.0 では偶数丸めが行われます。

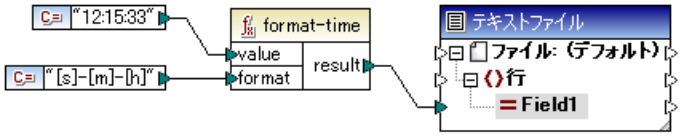
数値	書式文字列	結果
1234.5	#,##0.00	1,234.50
123.456	#,##0.00	123.46
1000000	#,##0.00	1,000,000.00
- 59	#,##0.00	- 59.00
1234	###0.0###	1234.0
1234.5	###0.0###	1234.5
.00025	###0.0###	0.0003
.00035	###0.0###	0.0004
0.25	#00%	25%
0.736	#00%	74%
1	#00%	100%
- 42	#00%	- 4200%
- 3.12	#.00;(#.00)	(3.12)
- 3.12	#.00;#.00CR	3.12CR

8.8.3.5 format-time

xs:time 入力値を文字列へ変換します。`format` 引数は **`format-dateTime`** 関数内の `format` 引数と同じ方法で使用されます。



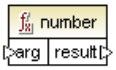
例



結果：33-15-12

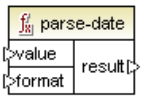
8.8.3.6 number

入力値の文字列を数値に変換します。また、ブール型入力も数値に変換します。



8.8.3.7 parse-date

Java、C#、C++、そして内蔵の実行エンジンで使用することができます。

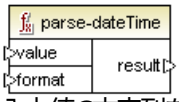


入力値のstring 文字列を日付に変更し、時刻に関する情報は無視されます。この関数はparse-dateTime 関数をベースにしたもので、その中でもtime コンポーネントが無視されます。出力はxs:date 型になります。

使用方法の例に関しては、次を参照してください：例 :RSS フィードからデータをマップする。

8.8.3.8 parse-dateTime

Java、C#、C++、そして内蔵の実行エンジンで使用することができます。



入力値の文字列をdateTime 型に変換します。

引数	説明
value	日付 /時刻を含む文字列。
format	与えられた値をフォーマットするために使用される文字列。

コンポーネント：

識別子	説明	デフォルトの表示
Y	年 絶対値）	4桁の数値 (2010)
M	その年の月	1- 12

識別子	説明	デフォルトの表示
D	その月の日付	1-31
d	その年の日付	1-366
H	時（24時間）	0-23
h	時（12時間）	1-12
P	A.M. または P.M.	アルファベット（言語に依存）
m	その時間の分	00-59
s	その分の秒	00-59
f	小数点以下の秒	数値、小数点1つ
Z	UTC からのタイムゾーンオフセット	+08:00
z	GMT からのタイムゾーンオフセット	GMT+n

書式修飾子：

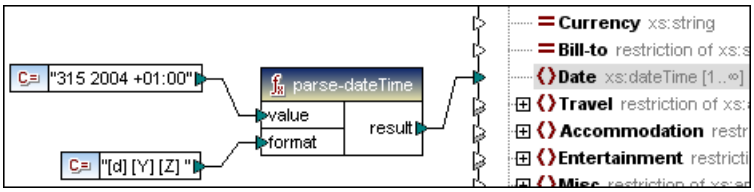
識別子	説明	サンプル
1	最初に 0を伴わない数値書式：1,2,3,...	1, 2, 3
01	2桁の数値書式：01、02、03、...	01, 02, 03
N	コンポーネント名（大文字）	MONDAY, TUESDAY
n	コンポーネント名（小文字）	monday, tuesday
Nn	コンポーネント（頭文字が大文字）	Monday, Tuesday

1) **N、n、Nn 修飾子は** M (month) コンポーネントでのみサポートされます。

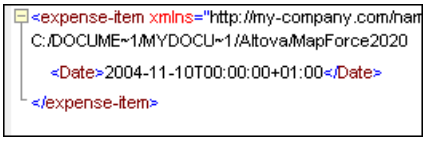
例：

日付文字列	書式文字列	結果
21-03-2002 16:21:12.492 GMT +02:00	[D]-[M]-[Y] [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. +01:00	[D].[MNn].[Y,2-2] [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

MapForce 内のサンプル：

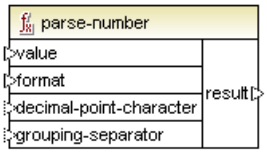


結果：



8.8.3.9 parse-number

Java、C#、C++、そして内蔵の実行エンジンで使用することができます。

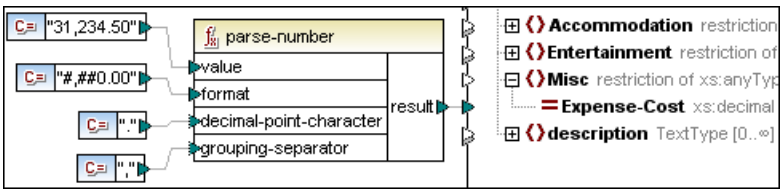


入力値文字列を数値に変換します。

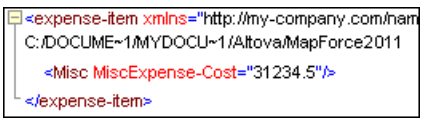
引数	説明
value	数値としてパース /変換する文字列。
format	数値を整形 (フォーマット) するためのフォーマット文字列。デフォルトは "###0#" になります。
decimal-point-character	小数点として使用される文字です。デフォルトでは "." になります (オプション)。
grouping-separator	数値グループを分割するのに使用されるセパレーター /デリミタです。デフォルトでは "," になります (オプション)。

parse-number のformat で使用される文字列は、format-number のそれと同じものです。

MapForce 内のサンプル:

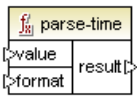


結果:



8.8.3.10 parse-time

Java、C#、C++、そして内蔵の実行エンジンで使用することができます。



入力値の文字列を時刻に変更し、日付に関する情報は無視されます。この関数はparse-dateTime 関数をベースにしたもので、その中でもdate コンポーネントが無視されます。出力はxs:time 型になります。

8.8.3.11 string

入力値を文字列型の出力値に変換します。関数はノードのテキストコンテンツとして使用することもできます。



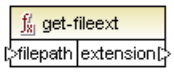
XML 複合型が入力ノードとなる場合、全ての子要素も単一のstring として出力されます。

8.8.4 core | file path functions

ファイルパス (file path) 関数を使用することで、ファイルパスのデータやファイルの拡張子へアクセスまたは操作することができ、更なるマッピング処理に使用することができます。これらの関数はMapForce にて使用可能な全ての言語で使用することができます。

8.8.4.1 get-fileext

ドット "."文字を含むファイルパスの拡張子を返します。



例 : c:\data\Sample.mfd' からは 'mfd' が返されます。

8.8.4.2 get-folder

ファイルパスから、フォルダーを分けるスラッシュやバックスラッシュ (日本語環境では円マーク) を含むフォルダー名を返します。



例 : c:/data/Sample.mfd' からは c:/data/' が返されます。

8.8.4.3 main-mfd-filepath

メインマッピングを含んでいるmfd ファイルのフルパスを返します。mfd ファイルが保存されていない場合、空の文字列が返されます。



8.8.4.4 mfd-filepath

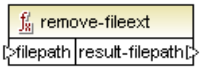
関数がメインマッピング内で呼び出された場合、main-mfd-filepath と同一の値（メインマッピングが含まれているmfd ファイルのフルパス）が返されます。mfd ファイルが保存されていない場合、空の文字列が返されます。



mfd ファイルへインポートされた**ユーザー定義関数**から関数が呼び出された場合、そのユーザー定義関数の**定義**を含んでいるインポートされたmfd ファイルのフルパスが返されます。

8.8.4.5 remove-fileext

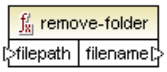
ドット文字を含むファイルパスの拡張子が削除されます。



例：'c:/data/Sample.mfd' からは 'c:/data/Sample' が返されます。

8.8.4.6 remove-folder

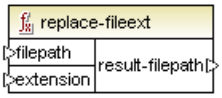
ファイルパスから、スラッシュやバックスラッシュ（日本語環境では円マーク）を含むフォルダー名が削除されます。



例：'c:/data/Sample.mfd' からは 'Sample.mfd' が返されます。

8.8.4.7 replace-fileext

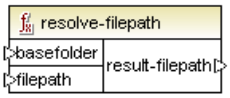
filepath パラメーターから得られたファイルパスの拡張子を、extension パラメーターから得られたファイルパスの拡張子で置き換えます。



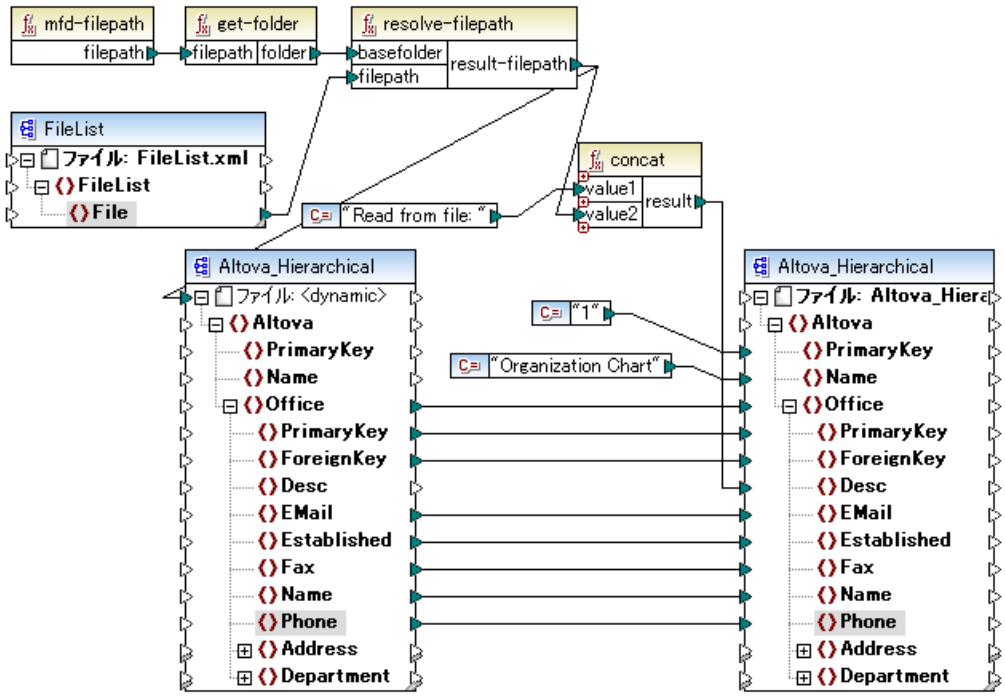
例：filepath 入力パラメーターに'c:/data/Sample.mfd' が与えられ、extension 入力パラメーターに 'mfp' が与えられた場合、'c:/data/Sample.mfp' が返されます。

8.8.4.8 resolve-filepath

相対ファイルパスから相対または絶対ベースフォルダーを解決します。関数では "." （カレントディレクトリ）と ".." （親ディレクトリ）を使用することができます。



..\MapForceExamples フォルダーに含まれている**MergeMultipleFiles_List.mfd** マッピングからサンプルを確認することができます。

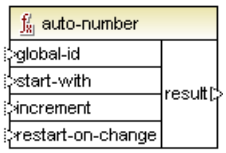


8.8.5 core | generator functions

core / generator 関数ライブラリは値を生成する関数を含みます。

8.8.5.1 auto-number

auto-number 関数により、指定されたパラメーターに従うかたちでコンポーネントのターゲットノードが統合されます。関数の結果は**start_with** から始まる値で、**インクリメント**で増加します。デフォルトの値 :開始=1、=1 づつ増加。双方のパラメータは負である可能性があります。

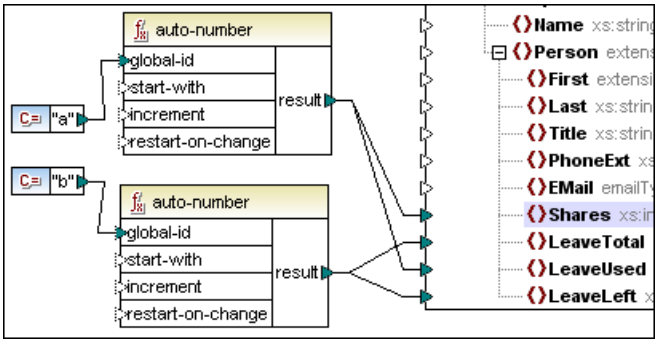


(auto-number 関数の) result コネクタがターゲットノードに**直接**接続されていることを確認してください。生成されたマッピングコードから関数が呼び出される正確な順番は定義されていません。MapForce ではキャッシュされた計算結果を再利用するか、任意の順序で条件式を再評価するかを選択することができます。従って、auto-number 関数を使用する際には特に注意する必要があります。

global-id
このパラメーターにより、独立した2つのauto-number 関数内に蓄えられているシーケンス出力の内容を同期することができます。

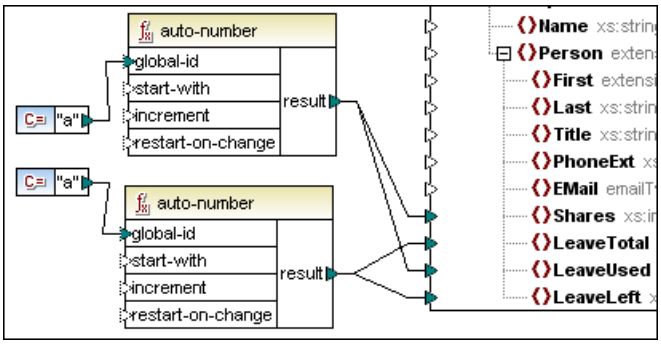
2 つのauto-number 関数が同一のglobal-id を**持たない**場合、各関数により別々の番号がターゲットアイテムにて追加されます。以下にあるサンプルでは、各関数に異なるgloba-id のa とb が与えられています。

マッピングの出力は 1、1、2、2 となります。上部にあるauto-number 関数により最初の 1が、下部のauto-number により2番目の 1が与えられます。



2 つの関数が**同一**のglobal-id である**a** を持っている場合、現在のauto-number の値が各関数により「共有」され、値が同期されることになります。

以下に示されるマッピングの出力は、1,2,3,4 となります。上部にある関数から最初のアイテムに 1が与えられ、2番目のアイテムには 2が与えられます。

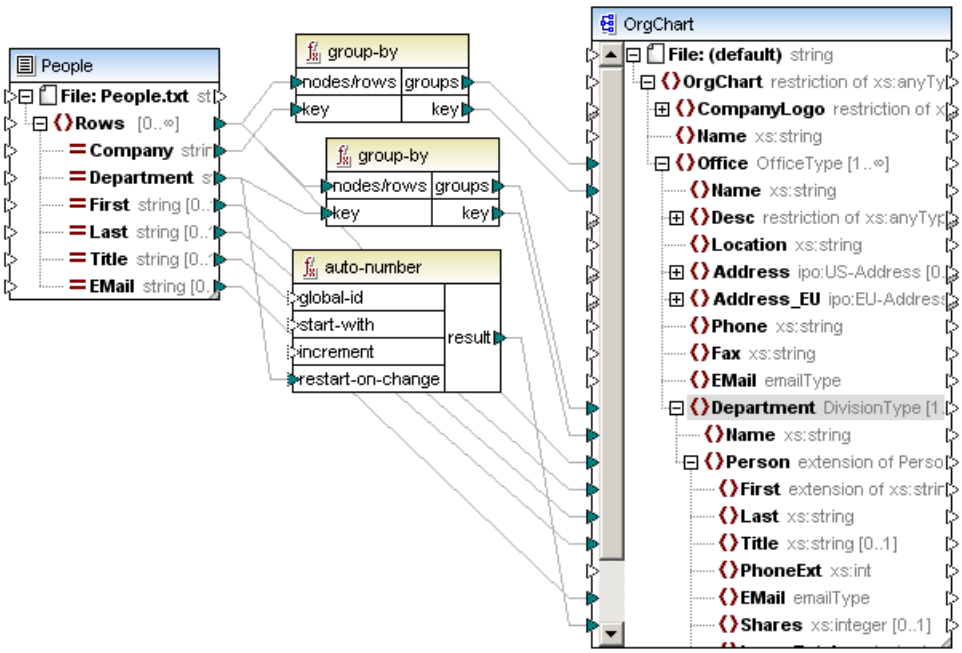


start-with
自動連番シーケンスの初期値として使用される値です。デフォルトでは 1が使用されます。

increment
シーケンスのインクリメントに使用される値です。デフォルトでは 1が使用されます。

restart on change
接続されているアイテムのコンテンツが変更された場合、自動連番のカウンターを "start-with" で指定された値にリセットします。

以下にあるサンプルでは、start-with とincrement の両方にデフォルトの 1が指定されています。Departmento のコンテンツが変更されるとつまり部門名が変更されると、連番に使用されているカウンターの値が 1にリセットされ、新たな部門に対して使用されるようになります。



8.8.6 core | logical functions

logical 関数は入力されたデータを比較して、"true" または "false" というboolean 型の出力を生成します。通常、フィルターを使用してサブセットを渡す前段階で、データを評価するために使用されます。

入力パラメーター = **a** | **b** または **value1** | **value2**
出力パラメーター = 結果

2 つの入力ノードの評価結果は、入力の値と比較に使用されたデータ型により異なります。

例 整数の値 4と 12の 'less than' 比較は、4が 12より小さいため、boolean 値 "true" を返します。 2つの入力文字列が '4'と '12'を含む場合、 '4'はアルファベットの順序でオペランド (12)の最初の文字 '1'よりも大きいため、構文的分析はの出力値 "false" を返します。

すべての入力データ型が同じ型の場合、 例 全ての入力ノードは数値型、または、文字列型であり、共通の型のための比較が行われます。

入力ノードが異なる型の場合、 例 整数と文字列 または 文字列と日付 そして、使用されるデータ型は、2つの入力型のうち **最も一般的な (最も制限のない)入力データ型です。**

2つの値を比較する前に、すべての入力の値は、共通のデータ型に変換されます。データ型 "文字列 " が "整数 "より制限のない前のサンプルを使用ます。整数の値 4を文字列 '12 'と比較し、整数の値 4を文字列 '4 'に変換し、文字列 '12 'と比較します。

メモ: Logical 関数をnull 値の存在をテストするために使用することはできません。null 値をlogical 関数の引数として与えると、null 値を返します。null 値の扱い方に関しての詳しい情報は、Nil 値 /Nillable を参照してください

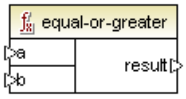
8.8.6.1 equal

a=b の場合、結果はtrue となり、それ以外の場合 false となります。



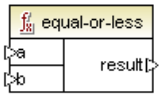
8.8.6.2 equal-or-greater

入力 a がb 以上の場合 true を返し、それ以外の場合にfalse を返します。



8.8.6.3 equal-or-less

入力 a がb 以下の場合 true を返し、それ以外の場合にfalse を返します。



8.8.6.4 greater

入力 a がb よりも大きい場合 true を返し、それ以外の場合にfalse を返します。



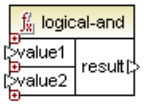
8.8.6.5 less

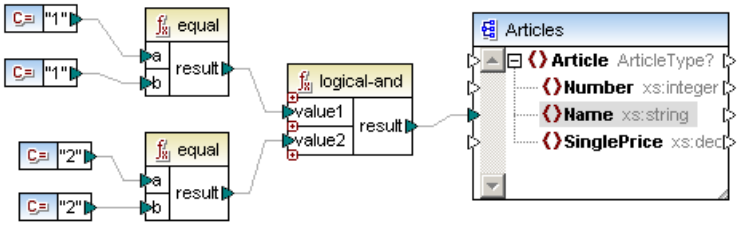
入力 a がb よりも少ない場合 true を返し、それ以外の場合にfalse を返します。



8.8.6.6 logical-and

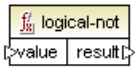
value1 とvalue2 の論理積が求められ、両方の値がtrue の時に、関数の結果もtrue となります。どちらかでもfalse の場合には結果もfalse となります。



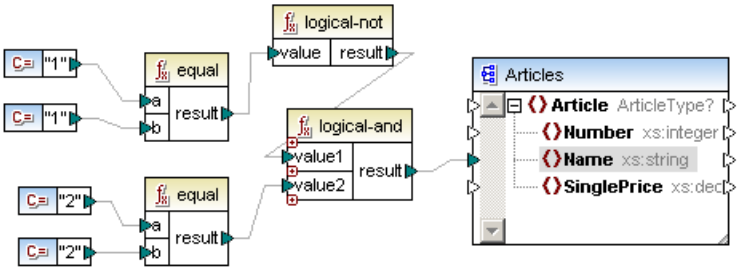


8.8.6.7 *logical-not*

論理状態 結果を反転します。入力がtrue であればlogical-not の出力はfalse になり、入力がfalse であれば出力がtrue になります。

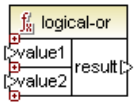


以下に示されるlogical-not 関数により、equal 関数の結果が反転されます。これにより、2 つあるequal 関数のどちらか一方がfalse で、もう一方がtrue になっている状態でしかlogical-and 関数からtrue が得られなくなります。



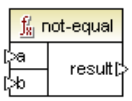
8.8.6.8 *logical-or*

両方の入力関数をboolean 型に揃える必要があります。value1 とvalue2 の少なくともどちらか一方がtrue となっている場合、関数の結果もtrue になります。両方の値がfalse の場合、結果もfalse になります。



8.8.6.9 *not-equal*

入力 a とb が等しく無い場合にtrue が返されます。

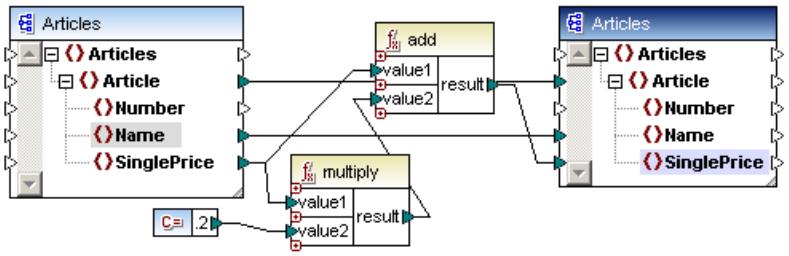


8.8.7 core | math functions

データを使った初歩的な計算を行うためにmath 関数を利用することができます。これら関数を使用して期間や日時の計算はできないという点に注意してください。

入力パラメーター = **value1** | **value2**
出力パラメーター = **result**

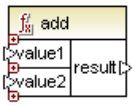
入力値は数値へ自動的に変換され、処理が行われます。



上に示されるサンプルマッピングでは、ターゲットコンポーネントにマッピングされた各製品の価格に対して20%の消費税が上乗せされます。

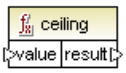
8.8.7.1 add

value1 と**value2** が加えられた数値が結果として返されます。



8.8.7.2 ceiling

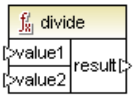
value にて与えられた値よりも大きな最小の整数値が返されます。つまり**value** にて与えられた数値の次に大きな整数が結果として返されます。



例えば、division 関数の結果が 11.2だった場合、ceiling 関数を使用することで、結果は (次に大きな整数値の) 12となります。

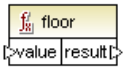
8.8.7.3 divide

value1 を**value2** で割った数が返されます。除算結果の精度はターゲット言語により変化します。round-precision 関数を使用することで、結果の精度を定義することができます。



8.8.7.4 floor

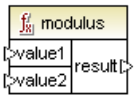
value にて与えられた値よりも小さな最大の整数値が返されます。つまり**value** にて与えられた数値の次に小さな整数が結果として返されます



例えば、division 関数の結果が 11.2だった場合、floor 関数を使用することで、結果は（次に小さな整数値の）11となります。

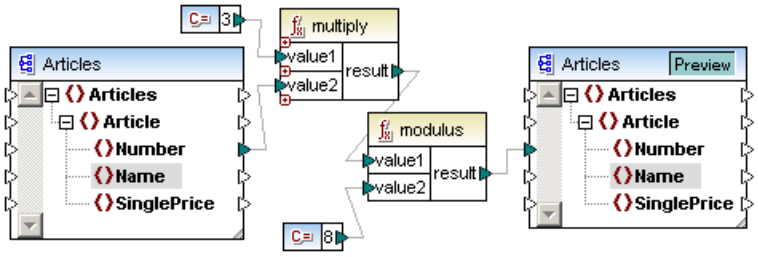
8.8.7.5 modulus

value1 を**value2** で割った値の余りが返されます。



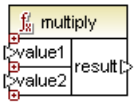
以下に示されるマッピングでは、Numbers の値が3倍されてmodulus 関数のvalue1 へ渡されています。入力値は 3、6、9、12となります。

modulus 関数のvalue2 入力に 8を与えると、余りとして 3、6、1、4が返されます



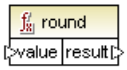
8.8.7.6 multiply

value1 と**value2** をかけ合わせた（乗算）値が返されます。



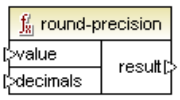
8.8.7.7 round

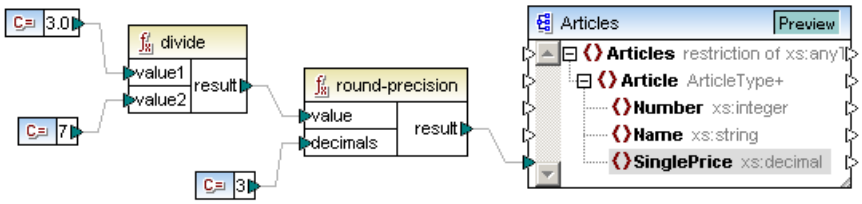
value により与えられた値の四捨五入結果が整数値で返されます。値が2つの整数の間の場合、"正の無限大への丸め"アルゴリズムが使用されます。例えば、"10.5"の値は "11"に四捨五入され、"-10.5"の値は "-10"に四捨五入されます。



8.8.7.8 round-precision

"decimals" により得られた値を小数点以下の有効数字として value 値の四捨五入を行います。

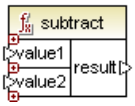




上のマッピングにより 0.429 という値が得られます。XML ファイルにて結果を正しく表示するには、xs:decimal 型の要素へマッピングするようにしてください。

8.8.7.9 subtract

value1 から**value2** を引いた数値を返します。



8.8.8 core | node functions

ノード関数を使用してノードにアクセスしたり、ノードを特定の方法で処理することができます。

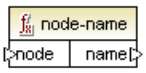
8.8.8.1 is-xsi-nil

ソースコンポーネントの要素ノードに xsi:nil 属性が含まれており、属性の値が "true" になっている場合、true (<OrderID>true</ OrderID>) を返します

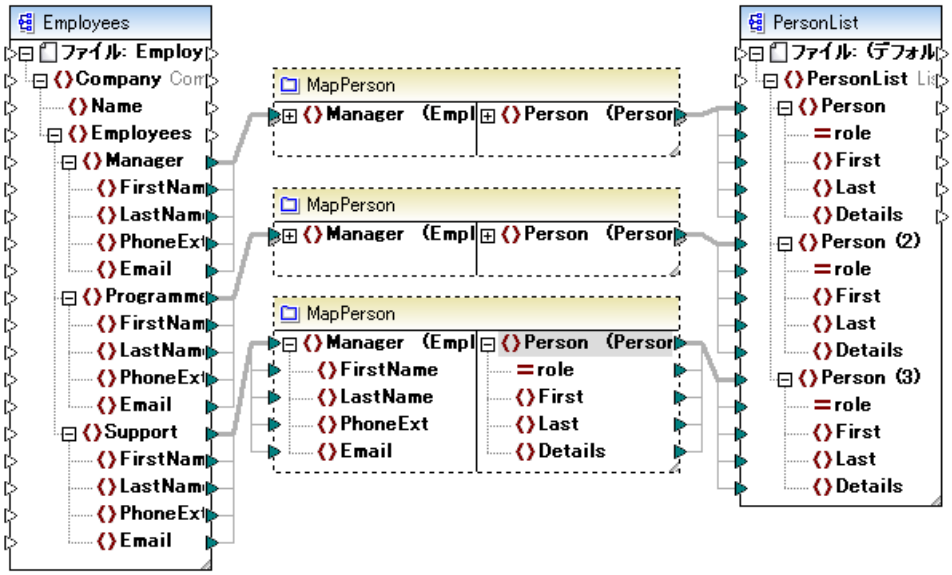


8.8.8.2 node-name

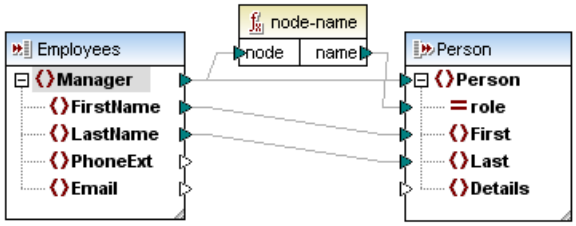
接続されているノードの修飾名 (QName) を返します。もしノードが XML text() の場合、空の QName が返されます。この関数は名前付きのノードに対してしか使用することはできません。(fn:node-name を呼び出す) XSLT をターゲット言語としている場合、関数は名前を持っていないノードに対して空のシーケンスを返します。



- データベースのテーブル/フィールドからの名前取得はサポートされていません。
- XBRL とExcel はサポートされていません。
- ファイル入力ノードの名前取得はサポートされていません。
- Web サービスのノードは、以下の例外を除き、XML ノードと同じような動作をします：
  - "part" からのノード名はサポートされていません。
  - ルートノード ("Output" または "Input") からのノード名はサポートされていません。



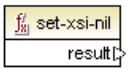
MapPerson のユーザー定義関数は**node-name** を使用することで入力**ノード**の名前を取得しており、その情報を role 属性に渡しています。ユーザー定義関数内にあるEmployees.xsd のルートノードは "Manager" として定義されています。



Manager はユーザー定義関数の**外側**からデータを取得します。データは "Manager"、"Programmer"、"Support" のどれかになります。このデータがPersonList のrole 属性に渡されます。

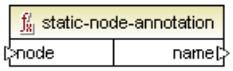
8.8.8.3 set-xsi-nil

ターゲットノードにxsi:nil をセットします。



8.8.8.4 static-node-annotation

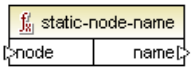
接続されたノードの注釈と共に文字列を返します。入力は以下である必要があります:(i) ソースコンポーネント ノード、または (ii) 呼び出しマッピング内のノードに直接接続されているパラメーターに直接接続されているインライン関数。



接続は直接である必要があります。フィルター、または非インラインユーザー定義関数をパススルーすることはできません。これは、接続されたノードから取得されたテキストの生成時に置き換えられた擬似関数で、全ての言語で使用することができます。

8.8.8.5 static-node-name

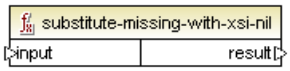
接続されたノードの名前と共に文字列を返します。入力は以下である必要があります:(i) ソースコンポーネント ノード、または (ii) an呼び出しマッピング内のノードに直接接続されているパラメーターに直接接続されているインライン関数。



接続は直接である必要があります。フィルター、または非インラインユーザー定義関数をパススルーすることはできません。これは、接続されたノードから取得されたテキストの生成時に置き換えられた擬似関数で、全ての言語で使用することができます。

8.8.8.6 substitute-missing-with-xsi-nil

ソースコンポーネントにて見つからない値 (またはnull 値) をxsi:nil 属性に置き換えます



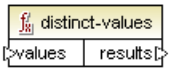
8.8.9 core | sequence functions

MapForce では入力シーケンスとそのコンテンツを処理するために用意されているsequence 関数を使用することができます。**key** 入力パラメーターの値 (コンテンツ) をノード行に接続することで、そのシーケンスをグループとして扱うことができます。

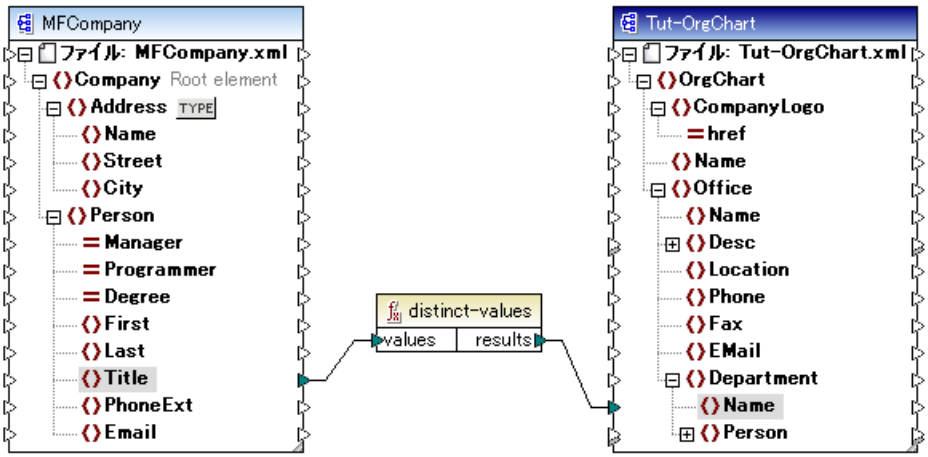
- 任意のデータ型に接続された入力パラメーター**key** の値は**group-adjacent** や**group-by** で使用できるようにstring 型へ変換されます。
- Boolean 型の**bool** 入力パラメーターは**group-starting-with** ならびに**group-ending-with** にて使用されます。
- 出力パラメーターの**key** は、現在のグループキーになります。

8.8.9.1 distinct-values

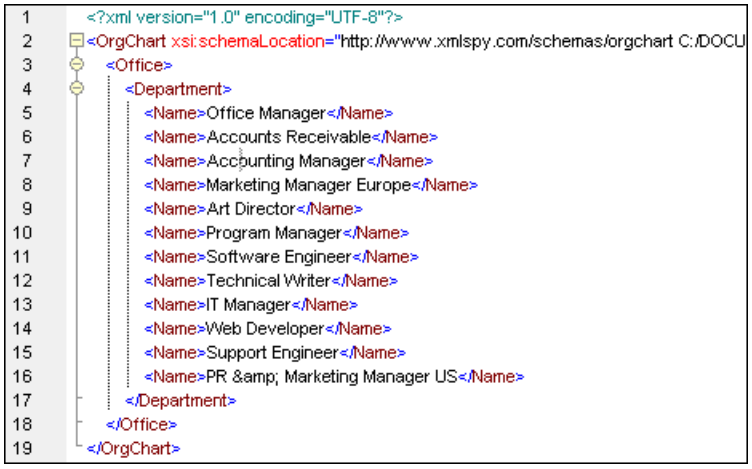
シーケンスにて重複した値を取り除き、ユニーク (一意) なアイテムをターゲットコンポーネントへ渡します。



以下の例では、ソースコンポーネントの "Title" アイテムにあるコンテンツがスキャンされ、一意（ユニーク）な役職だけが、ターゲットコンポーネントのDepartment / Name アイテムへマッピングされます。

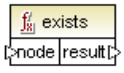


ソースコンポーネント内にあるTitle アイテムの順番はターゲットコンポーネントでも保持されていることに注目してください。



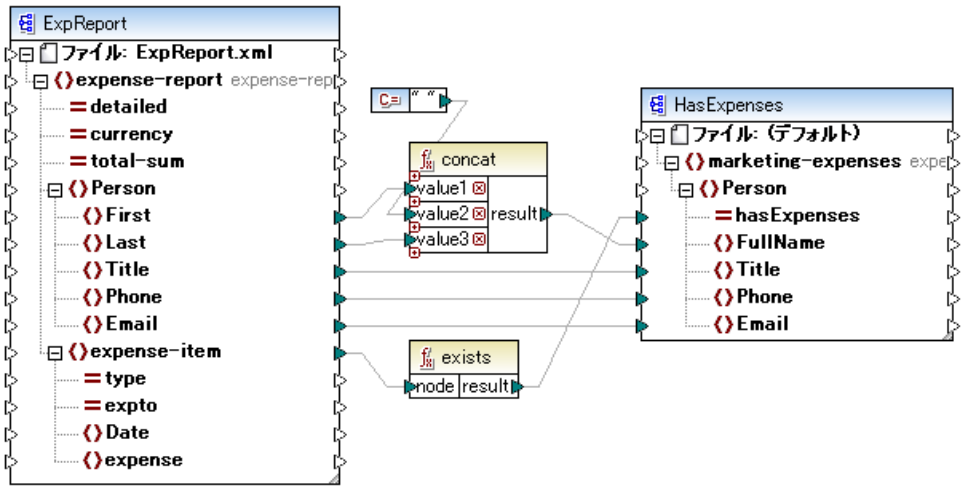
8.8.9.2 exists

ノードが存在する場合にtrue を返し、存在しない場合はfalse を返します。



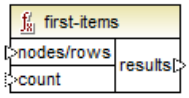
...\MapForceExamples フォルダー内部の**"HasMarketingExpenses.mfd"** ファイルには、以下に示されるサンプルが収められています。

expense-item がソースXML にて存在する場合、ターゲットXML/スキーマファイルの "hasExpenses" 属性値がtrue となります。



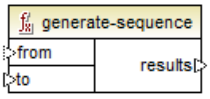
8.8.9.3 *first-items*

"count" パラメーターにより数値がX 提供される、入力シーケンスの最初の "X" アイテムを返します。例 :値 3がcount パラメーターとnodes/row パラメーターの親ノードにマップされ、最初の3つのアイテムが出力としてリストされます。



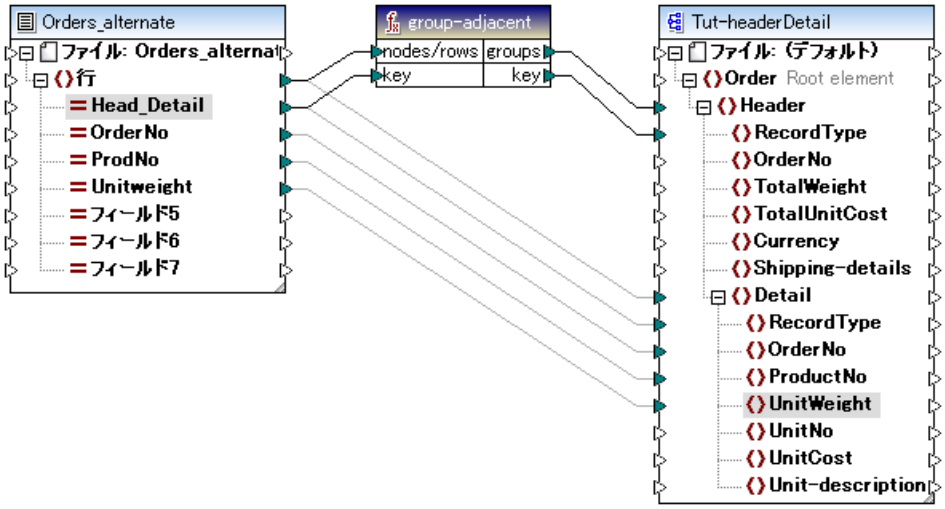
8.8.9.4 *generate-sequence*

"from" と "to" パラメーターを境界として使用し、整数のシーケンスを作成します。



8.8.9.5 *group-adjacent*

入力シーケンスのノード行を、同じ**キー**を持つ隣接したグループへ変換します。group-adjacent はグループ化のキーとしてノード/アイテムの**コンテンツ**を使用する点に注意してください



以下にあるCSV を、H (Header) とD (Detail) で、それぞれのグループに分けて収めるようにします。

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		

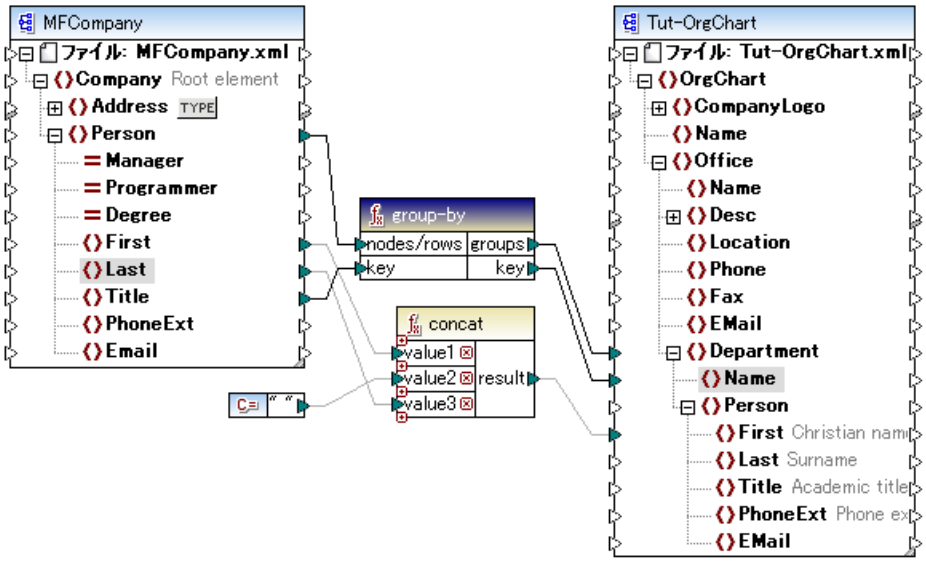
- 新しいグループが最初の要素 (この場合はH) にて構成されます。
- シーケンス内にある次の要素 (又はキー) は異なっている (D) ため、D という名前の新たなグループが構成されます。
- 次に来る 2 つのD 要素が、(同じ種類であるため) 同じD グループに追加されます。
- 次のH 要素により、新たなグループが作成されます。
- 最後にある 2 つのD 要素により、新たなD グループが作成されます。

```
<Header>
  <RecordType>H</RecordType>
  <Detail>
    <RecordType>H</RecordType>
    <OrderNo>111</OrderNo>
    <ProductNo>G-889-11</ProductNo>
    <UnitWeight>9</UnitWeight>
  </Detail>
</Header>
<Header>
  <RecordType>D</RecordType>
  <Detail>
    <RecordType>D</RecordType>
    <OrderNo>111</OrderNo>
    <ProductNo>B-192-409</ProductNo>
    <UnitWeight>1</UnitWeight>
  </Detail>
  <Detail>
    <RecordType>D</RecordType>
    <OrderNo>111</OrderNo>
    <ProductNo>F-152-427</ProductNo>
    <UnitWeight>3</UnitWeight>
  </Detail>
  <Detail>
    <RecordType>D</RecordType>
    <OrderNo>111</OrderNo>
    <ProductNo>Q-821-207</ProductNo>
    <UnitWeight>7</UnitWeight>
  </Detail>
</Header>
```

8.8.9.6 group-by

入力シーケンスのノード(行)を同じキーを持つグループへ変換します(グループが隣接しているとは限りません)。入力シーケンスにて出現する**キー**の順序がグループの出力に使用されます:

- ソースコンポーネントからグループ化を行うためのキーは Title アイテムとなります。このキーにより、company の従業員 (Person) がグループ化されます。
- ターゲットコンポーネントの Department/Name にグループ名が出力されるとともに、従業員の苗字 (Last) と名前 (First) が連結され、Person/First 子アイテムへ出力されます



group-by ではノード/アイテムの**コンテンツ**をグループのキーとして使用します！Title フィールドのコンテンツが従業員のグループ化、そしてターゲットにある Department/Name アイテムへのマッピングにて使用されます。

メモ：group-by 関数は、サンプルとそのマッピング結果からも分かる通り、ソースドキュメントからターゲットドキュメントへの**暗黙的なフィルタリング**としても機能します。ターゲットドキュメントでは、Person アイテムが、Title キーにより**グループ化**された各 Department アイテムに含まれています。

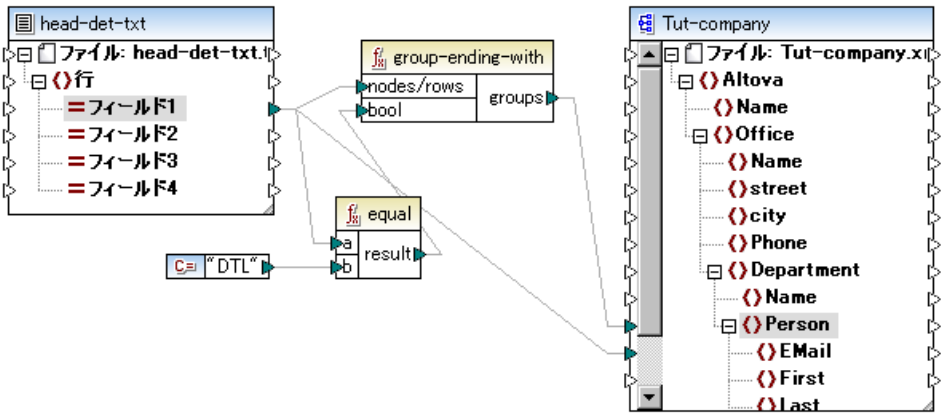
（CSV や FLF といった）フラットな構造のファイルがソースコンポーネントの場合、group-by 関数の 'key' 値を使用することで、動的な出力ファイル名を得ることができます。この場合、groups 出力をターゲットコンポーネントへ接続する必要はありません。\MapForceExamples フォルダーにある MultipleFilesFromTextfile.mfd を参照ください。

出力ボタンをクリックすることで、グループ化処理の結果を確認することができます。

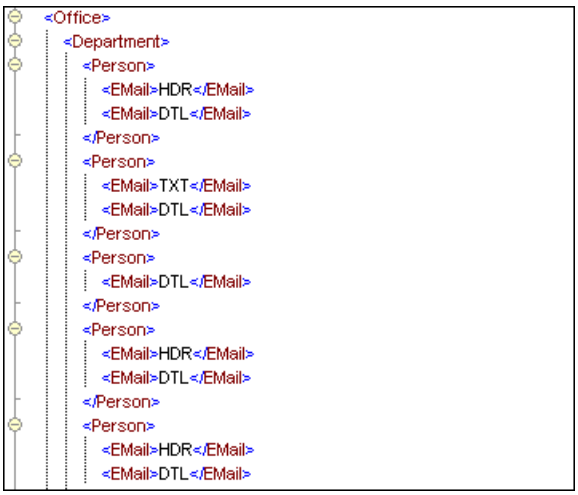
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3    <Office>
4      <Department>
5        <Name>Office Manager</Name>
6        <Person>
7          <First>Vernon Callaby</First>
8          <First>Steve Meier</First>
9        </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>
```

8.8.9.7 group-ending-with

この関数により入力シーケンスのノード/行から、bool 入力が true の時の入力が最後の要素となるグループが作成されます。このサンプルは "DTL" を group-ending-with アイテムとして使用した結果を表示しています。



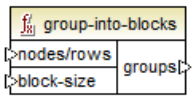
この場合、アイテム/ノードの**値**が一致または存在する必要は必ずしもありません。グループ化を行うためにはノード**パターン**（ノード/アイテム名）を一致させる必要があります。



上の結果から、DTL が最後の要素となるようなグループが作成されることが理解できます。

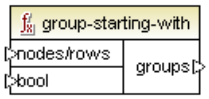
8.8.9.8 group-into-blocks

block-size パラメーターにより与えられた数値により定義された同じ値を持つ、入力シーケンス**nodes/rows** をブロックにグループ化します。

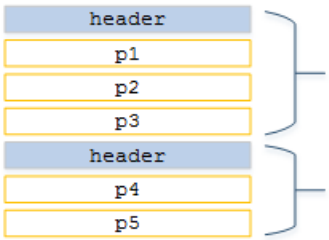


8.8.9.9 group-starting-with

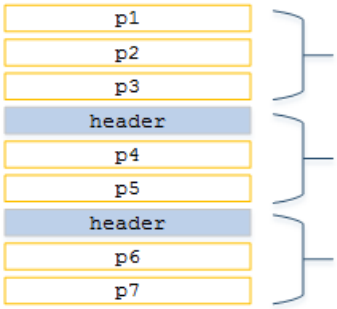
この関数により入力シーケンスのノード行から、**bool** 入力がtrue の時の入力が最初の要素となるグループが作成されます。



次のサンプルは、ノードのシーケンスがノードの "header" が発生し、**ブール値の** `true` を返す箇所を示しています。このノードのシーケンスに**`group-starting-with`** 関数を適用すると、結果として発生する2つのグループは、以下のとおりです。

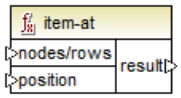


シーケンス内の最初のノードは、**ブール** の値に関係なく、新しいグループで開始することにご注意してください。すなわち、下のようなシーケンスは3つのグループを作成します。



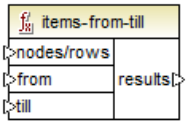
8.8.9.10 item-at

position パラメーターにより与えられたポジションにある**nodes/rows** を返します。 を返します。最初のアイテムは、ポジション "1"にあります。



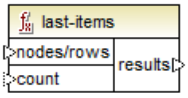
8.8.9.11 items-from-till

"from" と "till" パラメーターを境界として使用し、**nodes/rows** のシーケンスを返します。最初のアイテムは、ポジション "1" にあります。



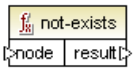
8.8.9.12 last-items

"count" パラメーターにより与えられた数値が X である場所のシーケンスの最後の "X" **nodes/rows** を返します。最初のアイテムは、ポジション "1"にあります。



8.8.9.13 not-exists

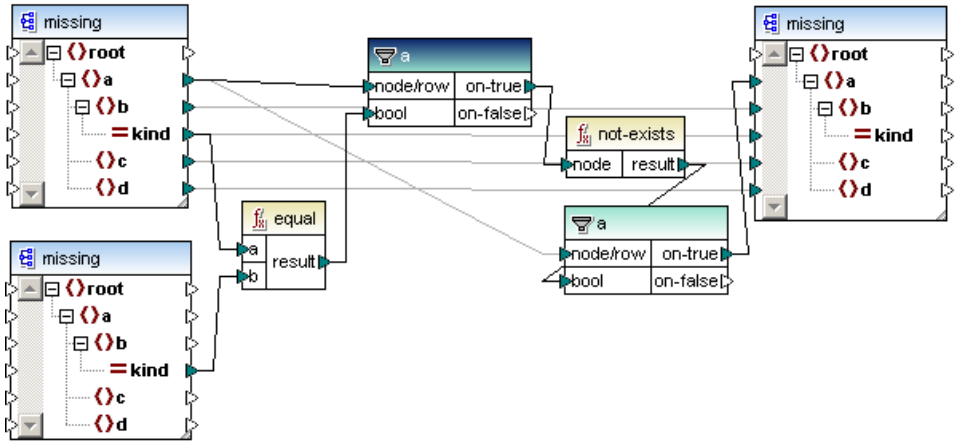
ノードが存在する場合 false を返します。



以下の例では、not-exists 関数を使用することでソースファイルには存在しないノードのマッピングを行います。

このマッピングでは、以下の処理を行います

- 2 つのソースXML ファイルにあるノードを比較します。
- 2 番目のソースXML ファイルに存在しないノードを、1番目のソースXML ファイルからフィルタリングします。
- 見つからないノードとそのコンテンツだけをターゲットファイルへマッピングします。



以下に 2 つのXML インスタンスを示します。以下の点が互いに異なっています：

- 左側にある**a.xml** では、b.xml には無い <b **kind="3"**> が含まれています。
- 右側にある**b.xml** では、a.xml には無い <b **kind="4"**> が含まれています。

```
<root xmlns:xsi="http://www.w3.
    <a>
        <b kind="1">b1</b>
        <c>c1</c>
        <d>d1</d>
    </a>
    <a>
        <b kind="2">b2</b>
        <c>c2</c>
        <d>d2</d>
    </a>
    <a>
        <b kind="3">b3</b>
        <c>c3</c>
        <d>d3</d>
    </a>
</root>
```

```
<root xmlns:xsi="http://www.w3.
    <a>
        <b kind="1">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
    <a>
        <b kind="2">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
    <a>
        <b kind="4">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
</root>
```

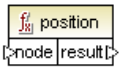
- equal 関数により、両方のXML ファイルにあるkind 属性が比較され、結果がフィルターコンポーネントへ渡されます。
- non-exists 関数が最初のフィルターの後に配置されており、各ソースファイルで見つからないノードを選択します。
- 2 番目のフィルターは、a.xml から得られた見つからないノードとそのデータだけを渡すためだけに使用されます。

マッピングにより、b.xml には存在しないノード <b kind="3"> がターゲットコンポーネントへ渡されます。

```
<root xsi:noNamespaceSchemaLocation="C:/DOCUME
    <a>
        <b kind="3">b3</b>
        <c>c3</c>
        <d>d3</d>
    </a>
</root>
```

8.8.9.14 position

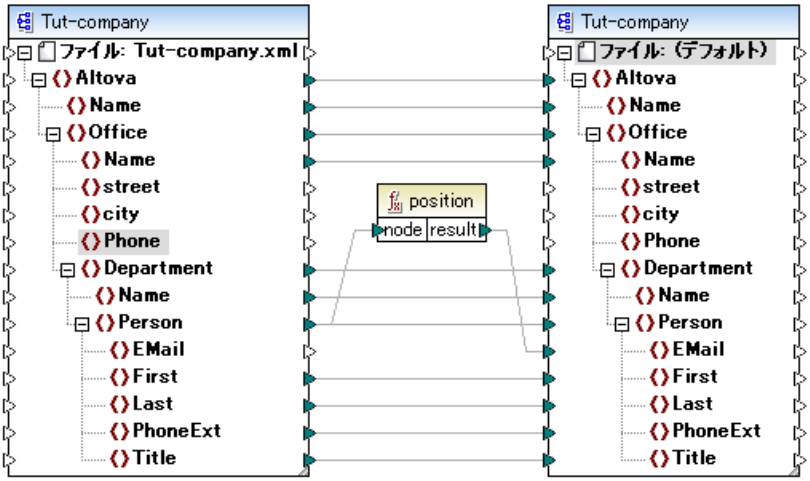
シーケンス内にあるノードの位置を返します。



position 関数を使用することで、シーケンス内部にある特定ノードの位置を決定したり、位置を元にしたフィルタリングを行うことができるようになります。

position 関数の node パラメーターへアイテムを接続することで、コンテキストアイテムが定義されます（以下の例では Person ノード）。

以下のマッピングでは、各部署（Department）の従業員（Person）に対して、（ソース XML にある要素の）位置番号が追加されます。



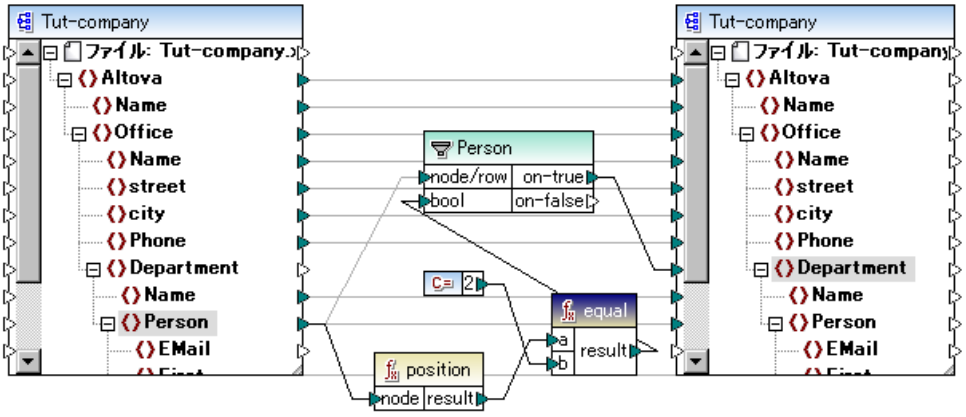
位置番号は、各部署ならびに各オフィスごとにリセットされます。

```
<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>
```

ポジション関数は特定のノードをフィルターするために使用されます。

ポジション関数はフィルターを使用して、ソースコンポーネント内に特定のポジションを持つ特定のノードをフィルターと共にマップすることができます。

フィルター "node/row" パラメーターとポジション "ノード" は、そのシーケンスの特定のポジションをフィルターアウトするために、同じソースコンポーネントのアイテムに接続されている必要があります。



このマッピングは以下を出力します：

- 各Department 内の第 2番目のPerson
- Altova 内の各オフィス.

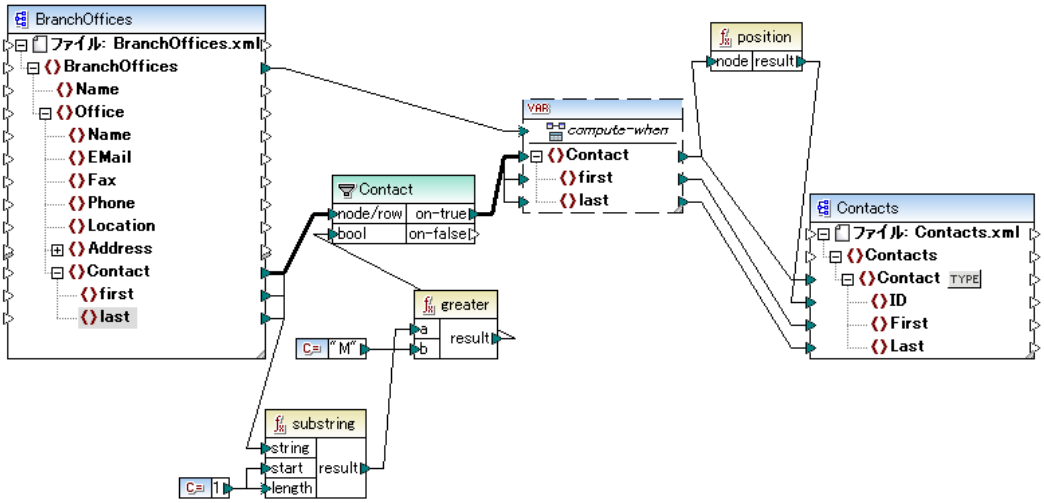
```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>b.bander@microtech.com</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
  <Department>
    <Name>Sales and Marketing</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>
```

フィルターされたシーケンス内のアイテムの位置を検索する：

フィルターコンポーネントはシーケンス関数ではないため、フィルターされたアイテムのポジションを検索する関数と共には position 関数と共に直接使用することはできませんこれを行うには、"変数" コンポーネントを使用します。

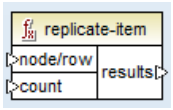
変数コンポーネントの結果は常にシーケンスです。例：シーケンスの作成のために使用することのできる区切られた値のリスト。

- 苗字 (Last) の頭文字がM より後 (つまりN 以降) の文字で始まるContact のデータがフィルタリングされ、変数コンポーネントへ集められます。
- これらのContact 情報が 変数コンポーネントから ターゲットコンポーネントへ渡されます。
- position 関数により、フィルタリングされた順にデータが番号付けされます。



8.8.9.15 replicate-item

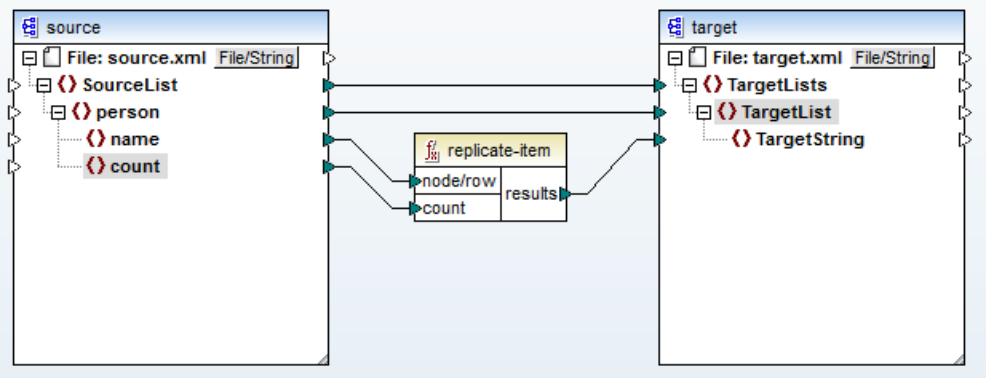
count 引数で指定された回数入力シーケンス内のアイテムを繰り返します。単一のアイテムをnode/row 引数に接続すると、関数は、N がcount 引数の値である箇所のN アイテムを返します。アイテムのシーケンスをnode/row 引数に接続すると、1度に1つのアイテムを処理し、関数は、シーケンス内のそれぞれのアイテムをcount 回繰り返します。例：count が 2の場合、シーケンス(1,2,3) は、(1,1,2,2,3,3)を生成します。



各アイテムのために異なったcount 値を与えることができる点に注意してください。各アイテムのために異なった値を与えることができることに注意してください。例 :以下の構造を持つソースXML ファイルがあると仮定します ：

```
<?xml version="1.0" encoding="UTF-8"?>
<SourceList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="source.xsd">
   <person>
      <name>Michelle</name>
      <count>2</count>
   </person>
   <person>
      <name>Ted</name>
      <count>4</count>
   </person>
   <person>
      <name>Ann</name>
      <count>3</count>
   </person>
</SourceList>
```

replicate-item 関数を使用して、ターゲットコンポーネント内で何度も個人を異なった回数繰り返し使用することができます。これを達成するには、 **replicate-item** 関数のcount 入力に各個人の<count> ノードを接続します：

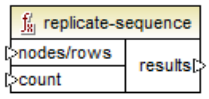


出力は、以下のとおりです：

```
<?xml version="1.0" encoding="UTF-8"?>
<TargetLists xsi:noNamespaceSchemaLocation="target.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <TargetList>
      <TargetString>Michelle</TargetString>
      <TargetString>Michelle</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
   </TargetList>
</TargetLists>
```

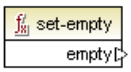
8.8.9.16 replicate-sequence

入力シーケンス内の全てのアイテムを `count` 引数内で指定された回数複製します。例：`count` が 2 の場合、シーケンス `(1,2,3)` は `(1,2,3,1,2,3)` を作成します。



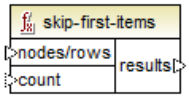
8.8.9.17 set-empty

空のシーケンスを返します。例えば、XBRL コンポーネントタクソノミの上位で定義された XBRL ドキュメントのデフォルトの値をキャンセルすることができます。



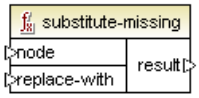
8.8.9.18 skip-first-items

X が "カウント" パラメーターにより提供される数量である、入力シーケンスの最初の "X" アイテム/ノードをスキップし、残りのシーケンスを返します。



8.8.9.19 substitute-missing

この関数は exists と if-else 条件の組み合わせにより構成されます。XML ソースファイルにてノードが存在する場合、現在のフィールドコンテンツがマッピングされ、それ以外の場合はマッピングされた "replace-with" パラメーターの内容が使用されます。

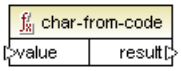


8.8.10 core | string functions

string 関数では、データの一部分を取得、文字列の検証、または文字列から必要な情報を検索するといった、様々な種類のソースデータを処理する為の文字列関数を使用することができます。

8.8.10.1 char-from-code

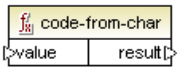
value により得られた整数値の Unicode 値を文字に変換した結果が返されます。



例に関しては以下を参照してください:特殊文字の置き換え。

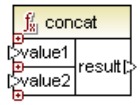
8.8.10.2 code-from-char

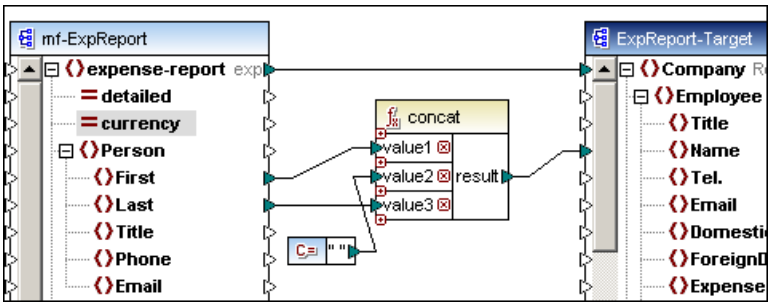
value から得られた最初の文字を整数値の Unicode へ変換した結果が返されます。



8.8.10.3 concat

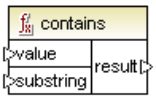
2 個以上の値を連結 (追加) して、result のパラメーターから出力します。全ての入力値は自動的に文字列型へ変換されます。





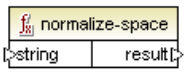
8.8.10.4 contains

substring パラメーターから得られた値が、value パラメーターの値に含まれる場合、true を返します。



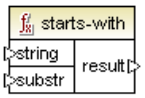
8.8.10.5 normalize-space

正規化された入力文字列が返されます。先頭と末尾にあるスペースが削除され、連続したスペースが 1 つのスペース文字に置き換えられます。Unicode 文字の空白スペースには (U+0020) が使用されます。



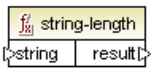
8.8.10.6 starts-with

string 入力パラメーターから得られた文字列が、**substr** パラメーターから得られた文字列で始まる場合、true を返し、それ以外の場合には false を返します。



8.8.10.7 string-length

string パラメーターにて得られた文字列の文字数を返します。



8.8.10.8 substring

string パラメーターから得られた文字列の一部分を返します。"start" パラメーターにより先頭文字の位置が指定され、"length" パラメーターにより、文字の長さが指定されます。

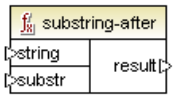


length パラメーターが指定されていない場合、start パラメーターにより指定された位置から、string パラメーターにより得られた文字列の最後までが返されます。インデックスは 1 から開始されます。

例：substring("56789",2,3) の結果は 678 となります。

8.8.10.9 substring-after

"**string**" パラメーターから得られた値の先頭から、**substr** パラメーターの値が、最初に出現するまでの箇所が削除されます。残りの文字列が関数の戻り値となります。**substr** が **string** 内で出現しなかった場合、空の文字列が返されます。

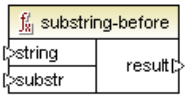


例 :substring-after("2009/01/04","/") の結果は、最初に出現する "/" より後の "01/04" となります。

使用方法の例に関しては、次を参照してください：例 :RSS フィードからのデータのマッピング。

8.8.10.10 substring-before

"**string**" パラメーターの先頭から、substr により指定された文字が出現するまでのフラグメントが返されます。**substr** が **string** 内で出現しなかった場合、空の文字列が返されます。



例：substring-after("2009/01/04","/") の結果は、最初に出現する "/" より後の "01/04" となります。

8.8.10.11 tokenize

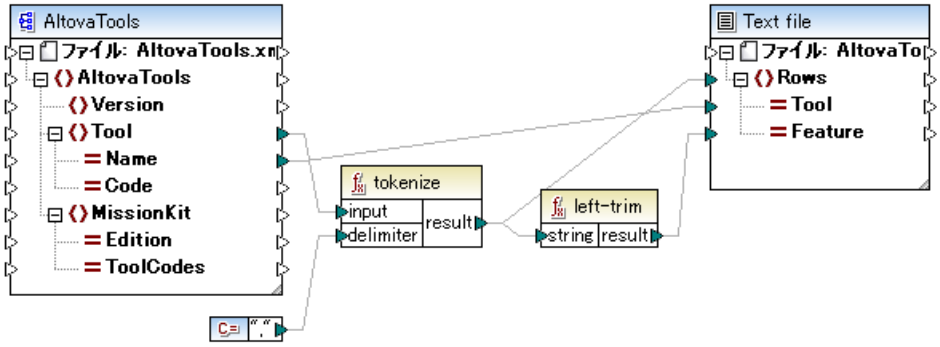
input パラメーターから入力文字列を delimiter パラメーターにより定義されたセグメントのシーケンスに分割します。関数の結果は更なる処理にて使用されます。



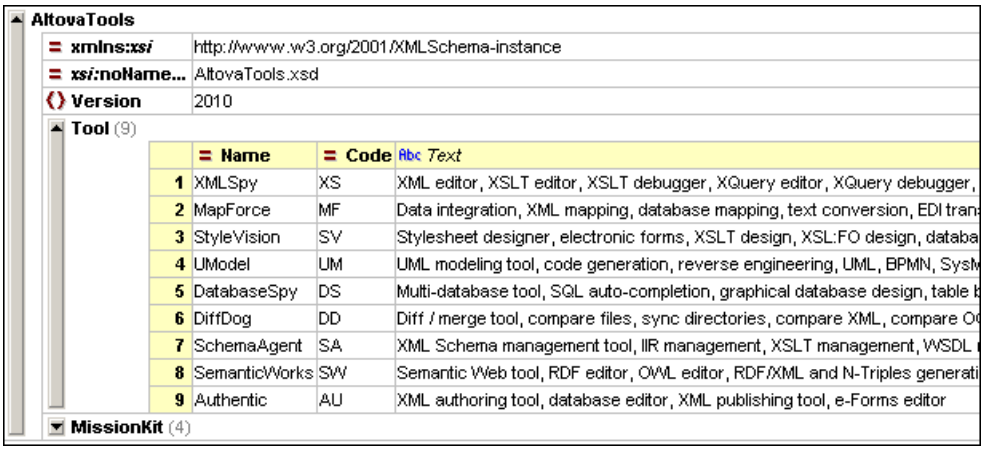
例：input の文字列が "A,B,C" で、delimiter から ","が与えられている場合、A B C がシーケンスとして出力されます。

サンプル

..\MapForceExamples フォルダーに収められている**TokenizeString1.mfd** ではtokenize 関数の使用方法を確認することができます。



XML ソースファイルを以下に示します。**Tool** 要素にはName とCode という2つの属性が含まれており、Tool 要素のコンテンツにはコマにより分けられたデータが与えられています。



AltovaTools			
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance		
xsi:noName...	AltovaTools.xsd		
Version	2010		
Tool (9)			
	Name	Code	Text
1	XMLSpy	XS	XML editor, XSLT editor, XSLT debugger, XQuery editor, XQuery debugger,
2	MapForce	MF	Data integration, XML mapping, database mapping, text conversion, EDI tran
3	StyleVision	SV	Stylesheet designer, electronic forms, XSLT design, XSL:FO design, databa
4	UModel	UM	UML modeling tool, code generation, reverse engineering, UML, BPMN, Sysl
5	DatabaseSpy	DS	Multi-database tool, SQL auto-completion, graphical database design, table
6	DiffDog	DD	Diff / merge tool, compare files, sync directories, compare XML, compare O
7	SchemaAgent	SA	XML Schema management tool, IIR management, XSLT management, WSDL
8	SemanticWorks	SW	Semantic Web tool, RDF editor, OWL editor, RDF/XML and N-Triples generati
9	Authentic	AU	XML authoring tool, database editor, XML publishing tool, e-Forms editor
MissionKit (4)			

マッピングの動作：

- tokenize 関数により **Tool** 要素 *アイテム*からデータが与えられ、コンマ ","デリミタにより 入力パラメーターにて与えられた文字列が分割されます 最初の要素は "XML editor" になります)。
- result パラメーターがターゲットコンポーネントの**Row** アイテムへマッピングされているため、分割された各要素ごとに新たな**行**が作成されます。
- result パラメーターは**left-trim** 関数にもマッピングされており、各要素にある先頭の空白スペースが取り除かれます。
- **left-trim** 関数のresult パラメーターは、ターゲットコンポーネントのFeature アイテムへマッピングされます。
- ターゲットコンポーネントとなっている出力ファイルはCSV ファイル (AltovaToolFeatures.csv) として定義されており、各フィールドはセミコロンにより区切られます (コンポーネントをダブルクリックすることで設定を確認することができます)。

マッピングの結果：

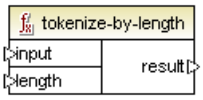
- ソースファイルの各 Tool 要素のため行われます。
- (Tool) 名がターゲットコンポーネント内のTool アイテムにマップされます。
- Tool コンテンツのトークン化された各チャンクが (Tool Name) Feature アイテムに追加さ
- 例 :最初のツールであるXMLSpy には、分割された最初の要素である "XML editor" が加えられます。
- このような動作が現在のTool と全てのTools に対して、繰り返されます。

- 出力タブをクリックすることで、以下にある結果を確認することができます。

```
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
```

8.8.10.12 tokenize-by-length

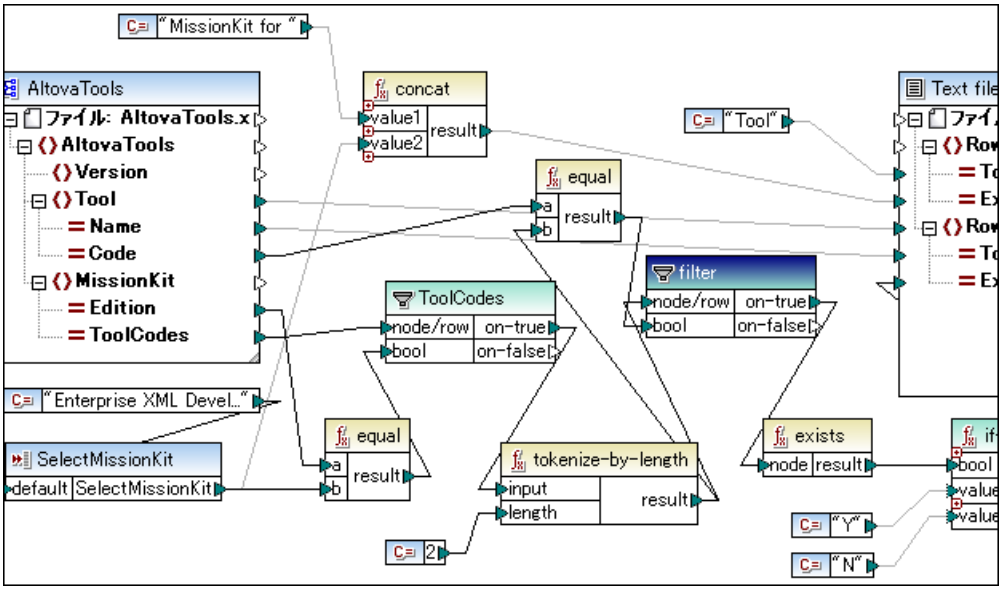
input パラメーターから得られた文字列をlength パラメーターにより定義されたセグメントのシーケンスに分割します。関数の結果は更なる処理にて使用されます。



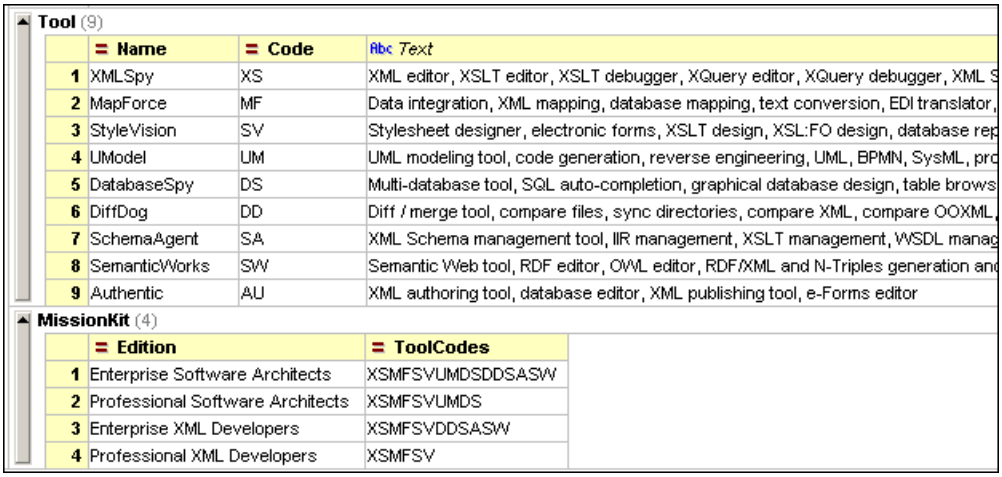
例 :input の文字列が "ABCDEF" で、length の値が 2 となっている場合、AB CD EF が出力されます。

サンプル

..\MapForceExamples フォルダーに収められている**TokenizeString2.mfd** にて、**tokenize-by-length** 関数の使用方法を確認することができます。



tokenize 関数の例で使用したものと同じXML ソースファイルを以下に示します。**MissionKit** 要素にも**Edition** と **ToolCodes** という 2 つの属性が与えられていますが、要素のコンテンツは無い点に注目してください。



Tool (9)

	Name	Code	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, prc
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	SW	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	XSMFSVUMDSDDSASW
2	Professional Software Architects	XSMFSVUMDS
3	Enterprise XML Developers	XSMFSVDDSASW
4	Professional XML Developers	XSMFSV

このマッピングの目的：
どのAltova 製品が、MissionKit の各エディションに含まれているかを表すリストを生成します。

マッピングの動作：

- SelectMissionKit **入力**コンポーネントには定数コンポーネントからデフォルト値（この場合は "Enterprise XML Developers"）が与えられます。
- **equal** 関数により、入力値と**Edition** コンポーネントの値が比較され、その結果がToolCodes フィルターの **bool** パラメーターへ渡されます。
- ToolCodes フィルターの**node/row** 入力がソースコンポーネントの**ToolCodes** アイテムから接続されています。例えばEnterprise XML Developers エディションの値はXSMFSVDDSASW となります。
- XSMFSVDDSASW という値が**on-true** パラメーターから、**tokenize-by-length** 関数の**input** パラメーターへ渡されます。

tokenize-by-length 関数の動作：

- ToolCodes から得られたXSMFSVDDSASW という**入力**値が、**length** パラメーターに接続されている 2 という値から、2文字ごとに複数の要素へ分割されます。従って合計 6個の要素が生成されます。
- 各要素はequal 関数のb パラメーターへ渡され、ソースファイルのCode に記述されている 2文字の**Code** 値と比較されます（Code には合計9個のエントリーアイテムが与えられています）。
- 比較の結果がフィルター（filter）の**bool** パラメーターへ渡されます。
- tokenize-by-length 関数から得られた**全て**の要素がフィルター（filter）の**node/row** パラメーターへ渡されている点に注意してください。

- **exists** 関数により、フィルターコンポーネントのon-true パラメーターから得られた値が存在する/存在しないかがチェックされます。

  存在するノードとは、Code の値と**ToolCodes** から得られた要素間でマッチするノードのことです。

  存在しないノードとはCode の値にマッチする**ToolCodes** の要素間でマッチしなかったノードのことです。

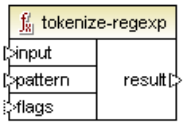
- **exists** 関数の結果は**if-else** 関数へ渡され、ノードが存在する場合にはY が、存在しない場合にはN がターゲットへ渡されます。

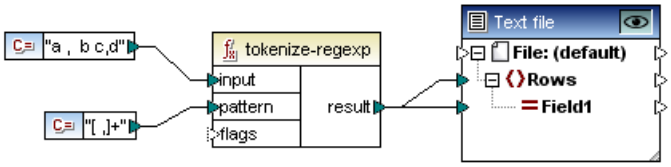
マッピング結果を以下に示します：

```
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
```

8.8.10.13 tokenize-regexp

input から得られた文字列を、文字列 (string) のシーケンスに変換します。**pattern** パラメーターにて得られた正規表現のパターンにマッチする文字列がセパレーターとして使用されます。セパレーターとして使用された文字列は関数の出力に表示されません。オプションの flags を使用することもできます。





上に示される例では：
空白スペースとコンマで区切られた文字列 (a , b, c,d) が**input** パラメーターから与えられています。

pattern に接続されている正規表現は [,] (["space""comma"]) となっており、空白スペースとコンマのどちらか一方がマッチする文字がセパレーターとして使用されます。

+ 文字を使用することで、「1 つ以上の」連続した文字クラス文字列の出現にマッチすることができます。

返される結果を以下に示します：

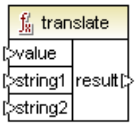
```
1    a
2    b
3    c
4    d
5
```

言語によって正規表現には細かな違いがある点に注意してください。C++ の Tokenize-regexp は Visual Studio 2008 SP1 以降でしか扱えない点に注意してください。

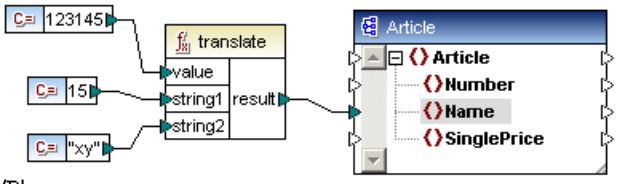
正規表現に関する更に詳しい情報については 正規表現 のセクションを参照ください。

8.8.10.14 translate

string1 (検索文字列) により得られた文字と同じ位置にある **string2** (置き換え文字列) の内容を使って "**value**" パラメーターの文字列を置き換えます。



string2 に対応する文字が無い場合、その文字は削除されることになります。



例：
入力文字列が "123145"で、
検索文字列）string1 ：15
置換文字列）string2 ：xy

となっている場合、
value パラメーターから得られた文字列にある "**1**" が **x** に置き換えられ、
value パラメーターから得られた文字列にある "**5**" が **y** に置き換えられます。

結果として得られる文字列は **x23x4y** となります。

string2 が空の場合（string1 よりも短い文字列の場合）、対応する文字列は削除されます。

例 2：
value から得られた入力文字列が "aabaacbca" で、
検索文字列）string1 は、 "a" です。
置換文字列）string2 は、 "" （空の文字列）です。

結果として "bcbc" が得られます。

例 3：
入力文字列が "aabaacbca" で、
string1 は、 "**ac**" です。
string2 は、 "**ca**" です。

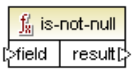
結果として "ccbccabac" が得られます。

8.8.11 db

db ライブラリは、データベース内にNULL フィールドが発生すると、マッピングの結果を定義することのできる関数を含みます。

8.8.11.1 is-not-null

field がNULL の時にfalse を返し、それ以外の場合にtrue を返します。



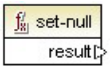
8.8.11.2 is-null

field がNULL の時にtrue を返し、それ以外の場合にfalse を返します。



8.8.11.3 set-null

データベースカラムにNULL をセットします。この関数ではデフォルト値のNULL も上書きされます。データベースフィールド以外のアイテムへ接続されている場合、空のシーケンスとして動作します。

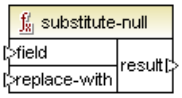


メモ:

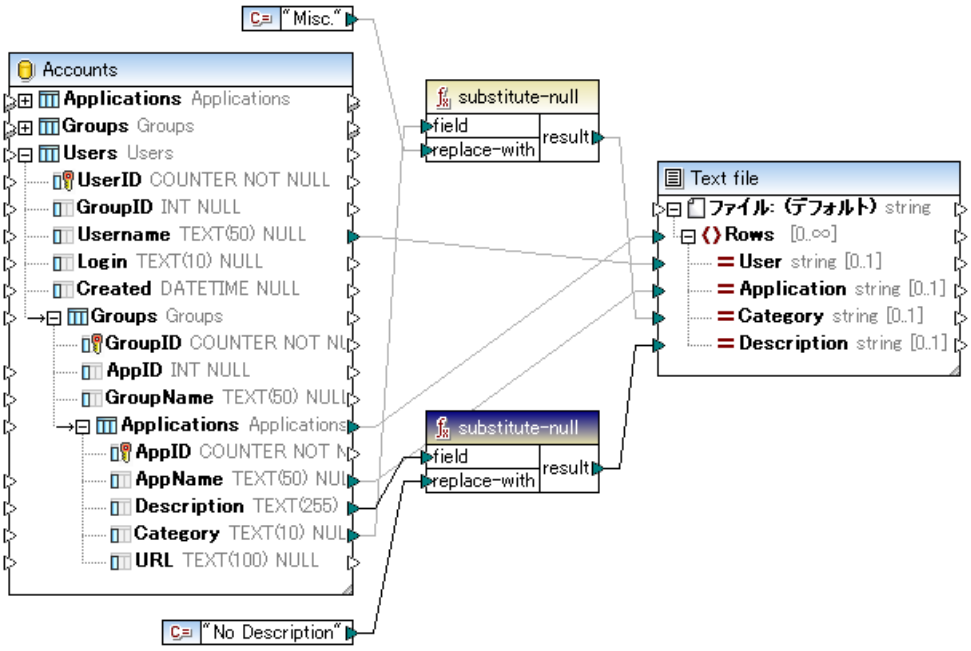
- 他の関数へset-null を接続しても、通常その関数は呼ばれません。
  例えばcount のような) シーケンス関数へset-null を接続すると、空のシーケンスが呼び出しに使用されます。
- 特別な関数、フィルターやIF-Else 条件へset-null を接続した場合、フィールドにNULL がセットされます。フィルターには"node/row" 入力を意味します。
- set-null から単純型への接続を行っても、ターゲットコンポーネント内に要素が作成されることはありません。

8.8.11.4 substitute-null

field パラメーターに接続されたアイテムのコンテンツが存在する場合、そのコンテンツがマッピングされ、それ以外の場合はreplace-with のパラメーターへ接続された内容がマッピングされます。



substitute-null 関数が実際に使用されている例を以下に示します。 **...\MapForceExamples** フォルダー内にある "**DB-ApplicationList.mfd**" を参照ください。



最初の関数により、Application テーブルにCategory エントリーが存在するかのチェックが行われます。エントリーが存在しない場合、"Misc." がテキストファイルのCategory アイテムへマッピングされます。

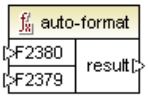
2 番目の関数では、説明 エントリーが存在するかのチェックが行われます。エントリーが存在しない場合には、"No description" という文字列がマッピングに使用されます。

8.8.12 edifact

edifact ライブラリには、EDI の日付、時刻、期間をxs:date 形式に変換するための関数が搭載されています。

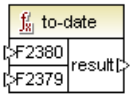
8.8.12.1 auto-format

与えられたフォーマットコード (F2379) に対して、コード化されたソース (F2380) から得られた日付 /時刻 / 日時の値を返します。



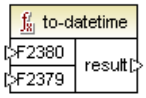
8.8.12.2 to-date

与えられたフォーマットコード (F2379) に対してコード化されたソース (F2380) から得られた日付 の値を返します。



8.8.12.3 to-datetime

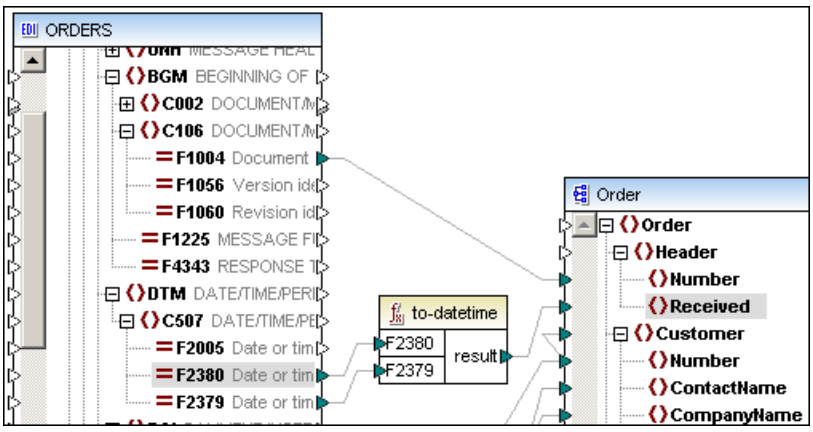
与えられたフォーマットコード (F2379) に対してコード化されたソース (F2380) から得られた日時の値を返します。



例：
以下のスクリーンショットに示されるEDI ファイルでは、DTM フィールドにて 日付 時刻 期間 に関する情報が与えられています。

```
UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
ADDR+970101:1230+00000000000001++ORDERS++++1'
UNH+0001+ORDERS:D:08A:UN'
BGM+221+ABC123456XYZ+9'
DTM+4:20040430174 2PDT:303'
FTX+PUR+3++Pizza purchase order'
RFF+CT:123-456'
RFF+CR:1122'
NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+C
CTA+SR+:Ted Little'
```

to-datetime 関数を使用することで、値がxs:date フォーマットへ変換されます。

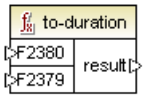


変換の結果を以下に示します。このサンプルは...\MapForceExamples の**EDI-Order.mfd** からも確認することができます。

```
<?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>
```

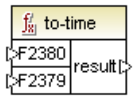
8.8.12.4 to-duration

与えられたフォーマットコード (F2379) を使用することで、コード化されたソース (F2380) から得られた期間の値を返します。



8.8.12.5 to-time

与えられたフォーマットコード (F2379) を使用することで、コード化されたソース (F2380) から得られた時刻の値を返します。

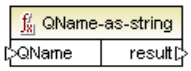


8.8.13 lang | QName functions

lang ライブラリには、Java、C#、またはC++ 言語のどれかが選択されている際に利用できる関数が収められています。

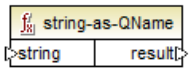
8.8.13.1 QName-as-string

QName 形式の一意な文字列が返されます。



8.8.13.2 string-as-QName

QName の文字列の表記をQName に戻します。

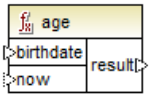


8.8.14 lang | datetime functions

lang ライブラリには、Java、C#、またはC++ 言語のどれかが選択されている際に利用できる関数が収められています。

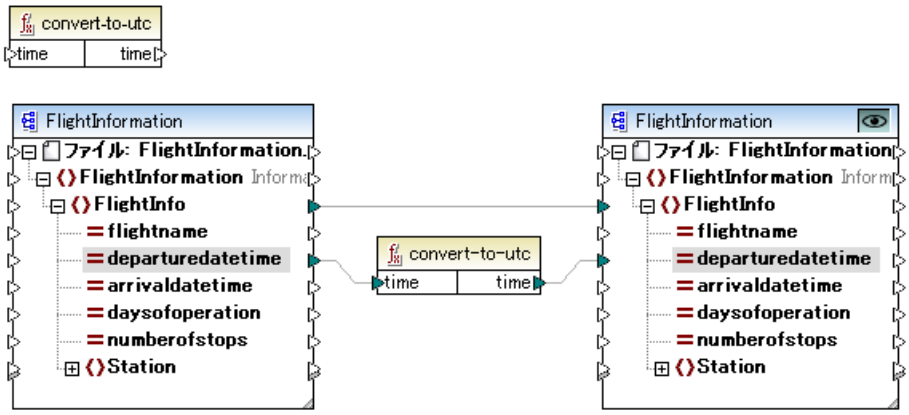
8.8.14.1 age

age
結果は個人の年齢を年数で表示します。*now* 引数は、任意でデフォルトは現在のシステムの日付です。結果は、生年月日から今日までの年数です。値が*now* 引数にマップされると、結果に、2つの日付の差異が表示されます。



8.8.14.2 convert-to-utc

ローカルのtime 入力パラメーターをUTC またはGMT/Zulu の時刻に変換します 関数の入力には +5:00 というタイムゾーンを表す記述を使用することができます)。



例 :インスタンスドキュメントにおける日時 :

```
departuredatetime="2001-12-17T09:30:02+05:00"
```

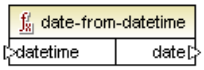
結果 :

```
departuredatetime="2001-12-17T04:30:02"
```

メモ:
ソースとなる日時が`departuredatetime="2001-12-17T09:30:02Z"` というフォーマットで記述されている場合、Zulu のタイムゾーンを表す **"Z"** が含まれているため、変換は行われません。結果は `departuredatetime="2001-12-17T09:30:02"` となります。

8.8.14.3 date-from-datetime

日時を表す入力パラメーターの日付部分を返します。T から始まるdateTime の時間部分はゼロとなります。タイムゾーンの内容は変更されない点に注意してください。



例 :インスタンスドキュメントにおける日時 :

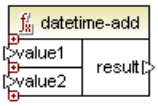
```
departuredatetime="2001-12-17T09:30:02+05:00"
```

結果 :

```
departuredatetime="2001-12-17T00:00:00+05:00"
```

8.8.14.4 datetime-add

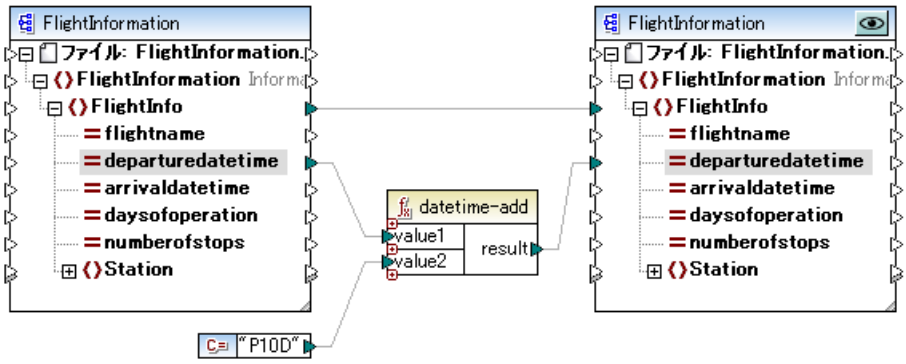
(2番目のパラメーターにより与えられる) 期間を 最初のパラメーターから得られた) 日時に加えて返します。



期間はP1Y2M3DT04H05M という形式で入力する必要があります。P の前にマイナス文字を加えることで、負の期間とすることができます (例： P1D)。

P は期間を表す文字で、必須の文字となります。上記の例にある期間は、 1年 2か月 3日、 時間を表すT をはさみ) 4時間 5分となります。

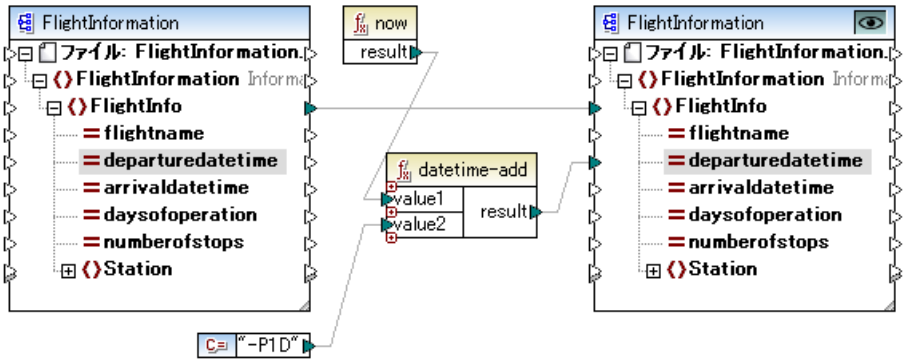
以下の例ではdeparturedatetime に 10日間 (P10D) が追加されます。



例 :インスタンスドキュメントにおける日時：
```
departuredatetime="2001-12-17T09:30:02+05:00"
```

結果：
```
departuredatetime="2001-12-27T09:30:02+05:00"
```

dateTime 入力にて昨日の日付を使用する：
now 関数を使用することで、タイムゾーンを含む現在の日時を入力します。P 文字の前にマイナス文字を使用することで、負の期間とすることができます。例えば P1D でマイナス 1日間となります。

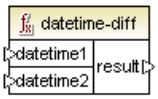


例 :現在時刻を表す now の値が 28th Feb 2012, 17:19:54.748(millisec)+01 である場合：
```
now="2012-02-28T17:19:54.748+01:00"
```

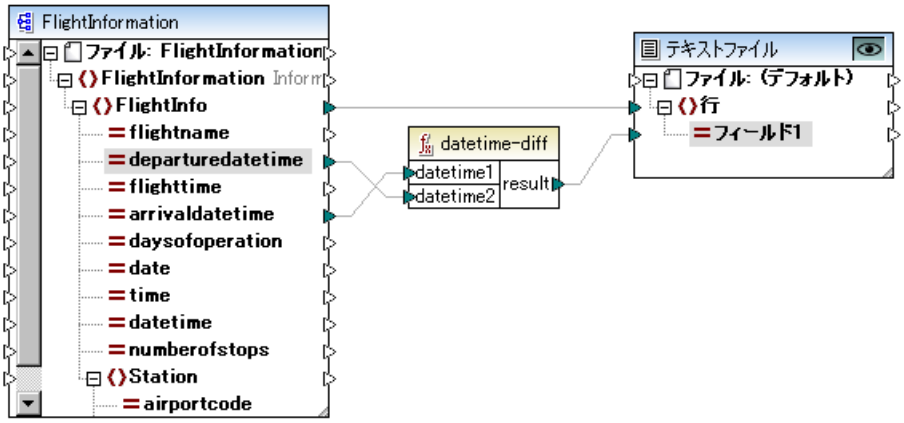
結果：
```
departuredatetime="2012-02-27T17:19:54.748+01:00"
```
つまり 2012年 2月 27日、17:19:54.748に、タイムゾーンの +01 が加わります。

8.8.14.5 datetime-diff

(2番目のパラメーターから得られる) datetime2 を (最初のパラメーターの) datetime1 から引くことで取得された期間を返します。結果は文字列型または duration データ型へマッピングすることができます。



arrivaldatetime が datetime1 へ、departuredatetime が datetime2 へ接続されていることにご注目してください。



例 :出発時刻 (departure time) と到着時間 (arrival time) との間にある時間を求めます。

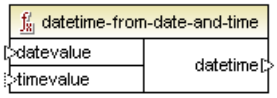
```
datetime1 arrivaldatetime="2001-12-17T19:30:02+05:00"
datetime2 departuredatetime="2001-12-17T09:30:02+05:00"
```

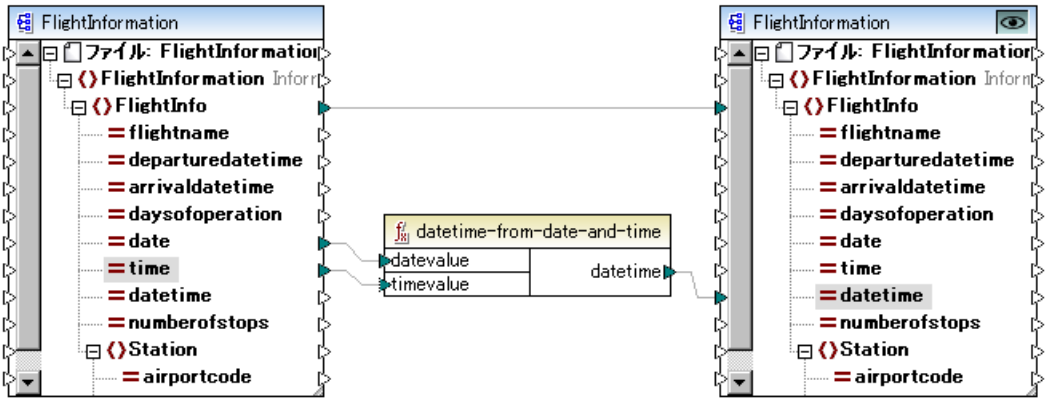
結果：これら 2 つの日時の差である10時間が得られます：

```
result= PT10H
```

8.8.14.6 datetime-from-date-and-time

(最初のパラメーターである) datevalue と (2番目のパラメーターである) timevalue から日付 (datetime) を構築します。最初のパラメーターは xs:date 型で、2番目のパラメーターは xs:time 型である必要があります。結果は文字列型または dateTime データ型へマッピングすることができます。



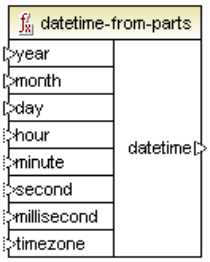


例：
```
date="2012-06-29"
time="11:59:55"
```

結果：
```
dateTime="2012-06-29T11:59:55"
```

8.8.14.7 datetime-from-parts

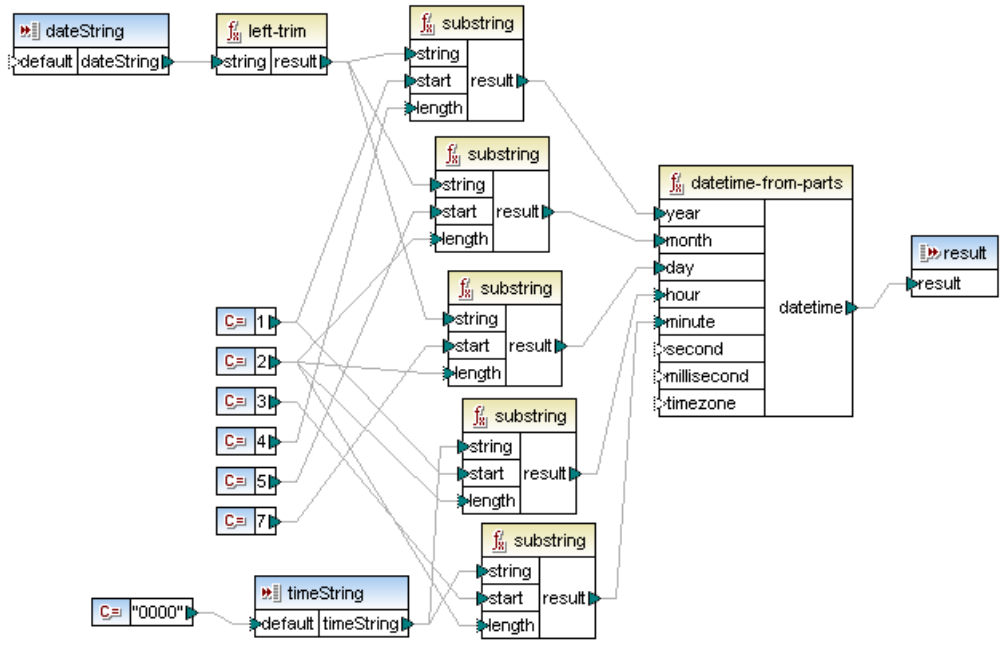
以下にある入力パラメーターから得られた値を組み合わせることで日時を構築します：year、month、day、hour、minute、second、millisecond、timezone。この関数では、得られた値が自動的に正規化され、例えば 1月 32日という値は 2月 1日という値に変更されます。



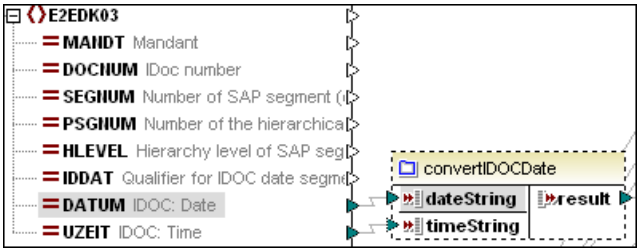
millisecond を除き、入力パラメーターの値は全て xs:int のデータ型となります。millisecond のデータ型は xs:decimal となります。関数から得られる結果のデータ型は xs:dateTime となります。

以下のスクリーンショットでは、..\MapForceExamples フォルダーに収められている IDoc_Order.mfd マッピングの convertIDOCDate ユーザー定義関数を示したものです。left-trim ならびに substring 関数を使用することで、入力文字列を組み合わせます。

year、month、day は必須のパラメーターで、それ以外のパラメーターはオプションとなっています。



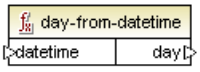
date ならびにtime フィールドはIDOC インスタンスファイルにて与えられます。

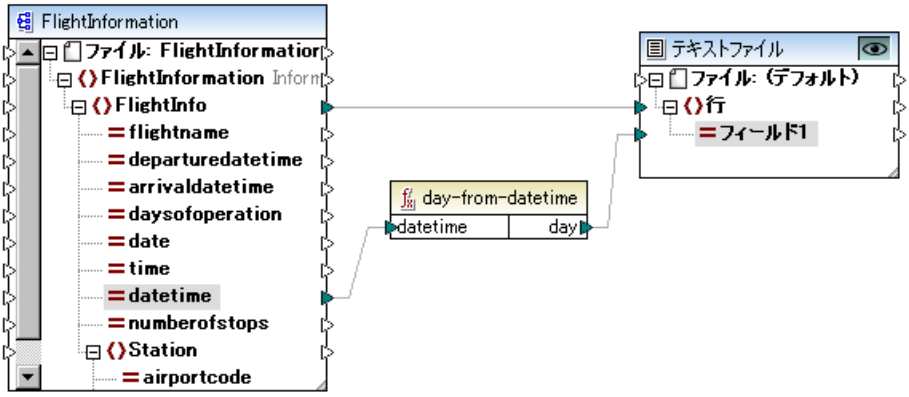



```
IDOC:Date    19990621
ICOC:Time    0930
Result       1999-06-21T09:30:00
```

8.8.14.8 day-from-datetime

datetime パラメーターの日付部分が返されます。

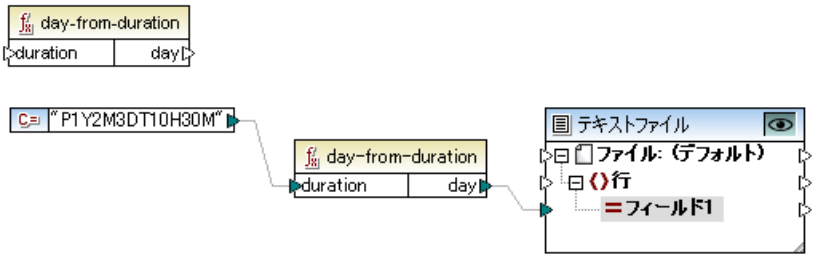




例：
```
datetime="2001-12-17T10:30:03+01:00"
```
結果：`17`

8.8.14.9 day-from-duration

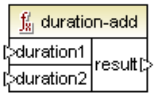
duration パラメーターの日付部分が返されます。



例：
```
duration="P1Y2M3DT10H30M"
```
結果：`3`

8.8.14.10 duration-add

2つの期間を足し合わせることで得られる期間を返します。



例：
```
duration1="P0Y0M3DT03H0M" (3days 3 hours)
duration2="P0Y0M3DT01H0M" (3days 1 hour)
```
結果：`P6DT4H (6days 4 hours)`

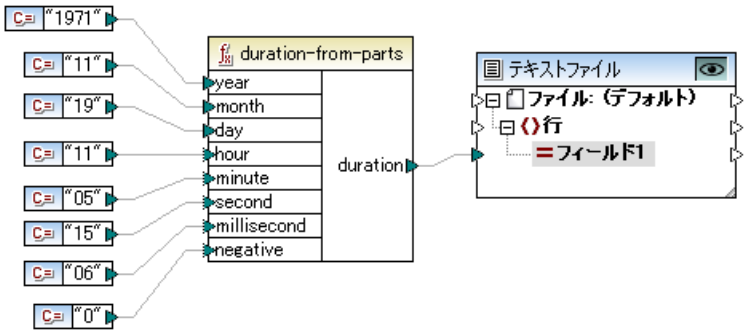
8.8.14.11 duration-from-parts

以下にある入力パラメーターから得られた値を組み合わせることで期間を構築します :year, month, day, hour, minute, second, millisecond, negative.



期間はP1Y2M3DT04H05M06 という形式を取り この場合はP(eriod) 1 年、 2か月、 3日、T(ime) 4 時 5 分 6秒 07ミリ秒という期間になります。

negative とmillisecond を除き 入力パラメーターの値は全てxs:int のデータ型となります。millisecond のデータ型はxs:decimal となり negative のデータ型はxs:boolean 型 (1でtrue、0でfalse) になります。関数から得られるduration のデータ型はxs:duration となります。



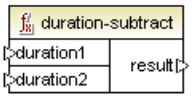
上の例では1971年、11月、19日、11時、 5分、15.06秒、そして負の期間を表すnegative の値は "false" となります。

結果：

```
duration="P1971Y11M19DT11H5M15.00006S"
```

8.8.14.12 duration-subtract

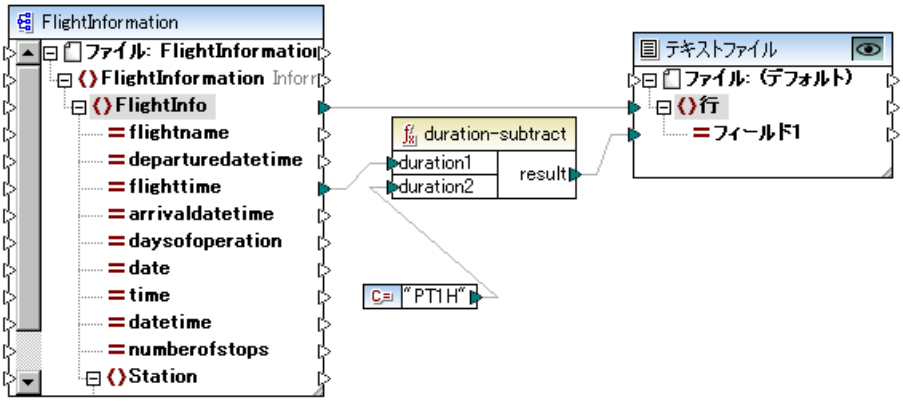
duration1 からduration2 を引いた値の期間を結果として返します。



入力パラメーターの値はP1Y2M3DT04H05M の形式で入力する必要があります。P 文字の前にマイナス文字を入力することで、負の期間を表すことができます (例： P1D)。

例ではP は期間を表す必須の文字で、 1年、 2か月、 3日、T(ime)、 4時間、 5分となります。

以下の例ではflighttime から 1時間 (PT1H) を引いた値が出力されます。

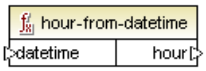


例：
```
duration1="P0Y0M0DT05H07M"
duration2="PT1H"
```

結果：`PT4H7M`

8.8.14.13 hour-from-datetime

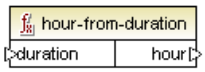
dateTime パラメーターの時間部分が返されます。



例：
```
datetime="2001-12-17T09:30:02+05:00"
hour= 9
```

8.8.14.14 hour-from-duration

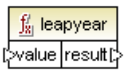
duration パラメーターの時間部分が返されます。

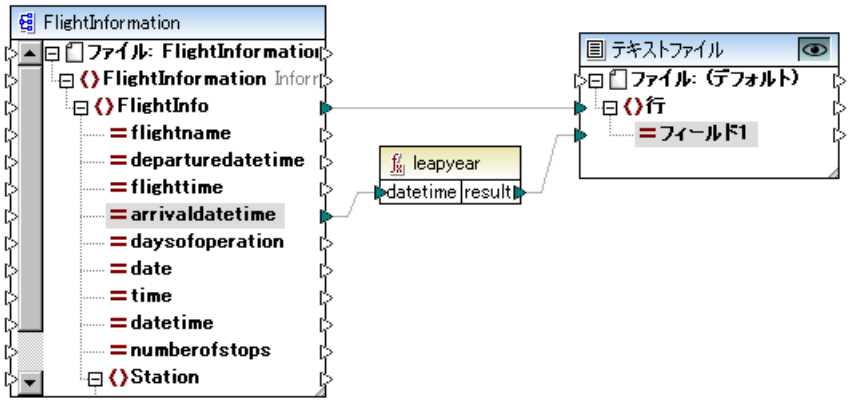


例：
```
duration="P0Y0M0DT05H07M"
hour= 5
```

8.8.14.15 leapyear

与えられたdateTime の年がうるう年である場合、true が返されます。



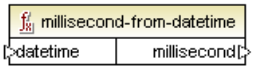


例：

```
arrivaldatetime="2001-12-17T19:30:02+05:00"
result="false"
```

8.8.14.16 millisecond-from-datetime

dateTime パラメーターのミリ秒部分が返されます。

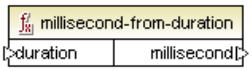


例：

```
datetime="2001-12-17T09:30:02.544+05:00"
millisecond= 544
```

8.8.14.17 millisecond-from-duration

duration パラメーターのミリ秒部分が返されます。

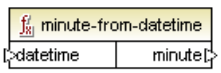


例：

```
duration="P0Y0M0DT05H07M02.227S"
millisecond= 227
```

8.8.14.18 minute-from-datetime

dateTime パラメーターの分部分が返されます。

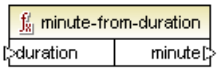


例：

```
datetime="2001-12-17T09:30:02.544+05:00"
minute= 30
```

8.8.14.19 minute-from-duration

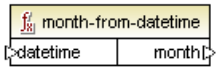
duration パラメーターの分の分部分が返されます



例：
```
duration="P0Y0M0DT05H07M02.227S"
minute= 7
```

8.8.14.20 month-from-datetime

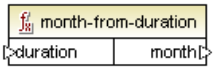
dateTime パラメーターの月の部分が返されます。



例：
```
datetime="2001-12-17T09:30:02.544+05:00"
month= 12
```

8.8.14.21 month-from-duration

duration パラメーターの月の部分が返されます。



例：
```
duration="P0Y04M0DT05H07M02.227S"
month= 4
```

8.8.14.22 now

(タイムゾーンを含む) 現在の日時が返されます。

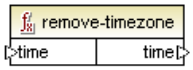


例：
```
result= 2012-03-06T14:44:57.567+01:00
```

1日前の値を得る方法については、core | lang | datetime-add 関数を参照ください。

8.8.14.23 remove-timezone

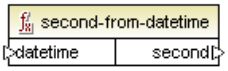
入力パラメーターからタイムゾーンコンポーネント (例：+5:00) を削除します。



例：
departuredatetime="2001-12-17T09:30:02**+05:00**"
time: 2001-12-17T09:30:02

8.8.14.24 second-from-datetime

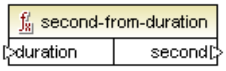
dateTime パラメーターの秒部分が返されます。



例：
datetime="2001-12-17T09:30:**02**.544+05:00"
second= 2

8.8.14.25 second-from-duration

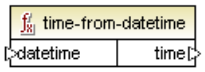
duration パラメーターの秒部分が返されます。



例：
duration="P0Y04M0DT05H07M**02**.227S"
second= 2

8.8.14.26 time-from-datetime

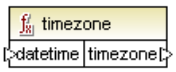
dateTime パラメーターの時間部分が返されます。



例：
datetime="2001-12-17T09:30:02.544+05:00"
time= 09:31:02+05:00

8.8.14.27 timezone

dateTime パラメーターの値から、UTC を基準にしたタイムゾーンが返されます。返されるtimezone の値の単位は分になります。



例：

```
datetime="2001-12-17T09:30:02.544+05:00"
timezone= 300
```

8.8.14.28 weekday

dateTime パラメーターから得られた値の曜日部分を返します。月曜日が1になり、日曜日が7になります。

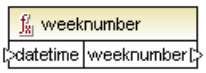


例：

```
datetime="2001-12-17T09:30:02.544+05:00"
weekday= 1
```

8.8.14.29 weeknumber

dateTime パラメーターにて得られた値から、その年の何番目の週かを返します。

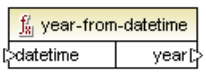


例：

```
datetime="2001-12-17T09:30:02.544+05:00"
weeknumber= 51
```

8.8.14.30 year-from-datetime

dateTime 引数にて得られた値の年部分を返します。

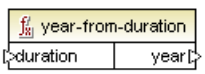


例：

```
datetime="2001-12-17T09:30:02.544+05:00"
year= 2001
```

8.8.14.31 year-from-duration

duration パラメーターにて得られた値の年コンポーネントを返します。



例：

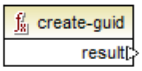
```
duration="P01Y04M0DT05H07M02.227S"
year= 1
```

8.8.15 lang | generator functions

lang ライブラリには、Java、C#、またはC++ 言語のどれかが選択されている際に利用できる関数が収められています。generator 関数により、スキーマやデータベース、またはEDI ソースコンポーネントからの入力を一切持たないデータベースフィールドの値が生成されます。

8.8.15.1 create-guid

特定のフィールドに対してグローバルに一意な識別子（16進数にエンコードされた文字列）が返されます。

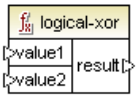


8.8.16 lang | logical functions

lang ライブラリには、Java、C#、またはC++ 言語のどれかが選択されている際に利用できる関数が収められています。

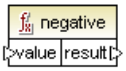
8.8.16.1 logical-xor

value1 の論理値とvalue2 の論理値が異なる場合にtrue を返し、それ以外の場合にfalse を返します。



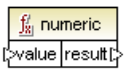
8.8.16.2 negative

値が否定を表す数値（-1以下）の場合 true が返され、それ以外の場合はfalse が返されます。



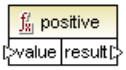
8.8.16.3 numeric

value の値が数値であればtrue を返し、それ以外の場合はfalse を返します。入力は通常文字列（string）となります。



8.8.16.4 positive

vaule の値が正の数であれば true を返します。0以下の整数では false が返されます。

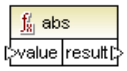


8.8.17 lang | math functions

lang ライブラリには、Java、C#、または C++ 言語のどれかが選択されている際に利用できる関数が収められています。

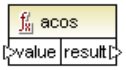
8.8.17.1 abs

value から得られた値の絶対値を返します。



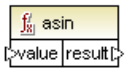
8.8.17.2 acos

value から得られた値のアークコサイン値を返します。



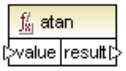
8.8.17.3 asin

value から得られた値のアークサイン値を返します。



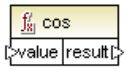
8.8.17.4 atan

value から得られた値のアークタンジェント値を返します。



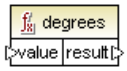
8.8.17.5 cos

value から得られた値のコサイン値を返します。



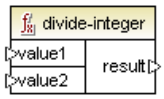
8.8.17.6 degrees

value から得られたラジアン値を度数へ変換した結果を返します。



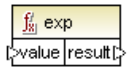
8.8.17.7 divide-integer

value1 の値を**value2** で除算した結果残る整数を返します。例えば 15 を 2 で割った結果は 7 となります。



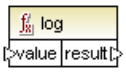
8.8.17.8 exp

自然対数の底である**e** を**value** から得られた値で階乗した結果が返されます。



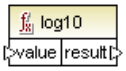
8.8.17.9 log

value から得られた値の対数が返されます。



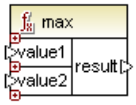
8.8.17.10 log10

value から得られた値の (底を 10 とする) 常用対数が返されます。



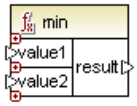
8.8.17.11 max

value1 と**value2** の値を数値として比較して、より大きな値が返されます。



8.8.17.12 min

value1 と**value2** の値を数値として比較して、より小さな値が返されます。



8.8.17.13 pi

円周率の値が返されます。



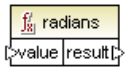
8.8.17.14 pow

a の値を**b** の値で**階乗**した結果が返されます。



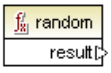
8.8.17.15 radians

value から与えられた度数をラジアン値に変換した結果を返します。



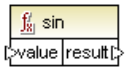
8.8.17.16 random

0.0 から 1.0 の間で擬似的なランダム値を返します。



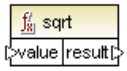
8.8.17.17 sin

value から得られた値のサイン値を返します。



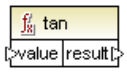
8.8.17.18 sqrt

value から得られた値の平方根を返します。



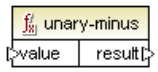
8.8.17.19 tan

value から得られた値のタンジェント値を返します。



8.8.17.20 unary-minus

符号付きの入力である**value** の値を反転させます (例：+3は -3 となり、-3は 3 となります)。

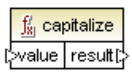


8.8.18 lang | string functions

lang ライブラリには、Java、C#、またはC++ 言語のどれかが選択されている際に利用できる関数が収められています。

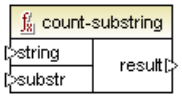
8.8.18.1 capitalize

value で与えられた文字列に含まれる各単語の頭文字を、大文字に変換します。



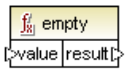
8.8.18.2 count-substring

substr にて指定された文字列が**string** 内に含まれる回数を返します



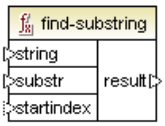
8.8.18.3 empty

入力文字列の**value** が空の場合にtrue を返し、それ以外の場合にfalse を返します。



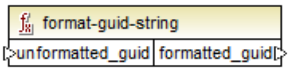
8.8.18.4 find-substring

string 文字列内でsubstr が最初に出現する位置を返します。startindex にて検索が開始される場所が指定されます。最初の文字の位置は 1 となります。substring が見つからなかった場合、0 が返されます。



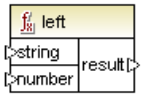
8.8.18.5 format-guid-string

unformatted_guid を入力文字列として、データベースのフィールドに使用するために正しくフォーマットされたGUID が**formatted_guid** から返されます。generator functions 以下にあるcreate-guid 関数の項目も参照ください。



8.8.18.6 left

number により指定された数の文字（列）を、**string** の文字列の先頭から取り出します。



例：string="This is a sentence" で、numer=4 の場合、"This" が返されます。

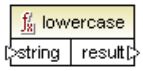
8.8.18.7 left-trim

与えられた文字列の先頭にある連続した空白スペースを削除して返します。



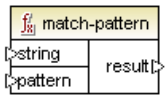
8.8.18.8 lowercase

入力パラメーターの**string** から得られた文字列を小文字に変換します。Unicode の文字の場合、(Unicode コンソーシアムにより定義された) 対応する小文字が使用されます。



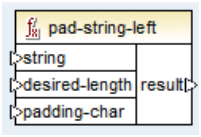
8.8.18.9 match-pattern

string パラメーターにより得られた文字列が、**pattern** にて定義された正規表現にマッチすればtrue を返し、それ以外の場合にはfalse を返します。細かな正規表現の構文は、選択されているターゲット言語に依存します。正規表現に関する詳細については正規表現 のセクションを参照ください。



8.8.18.10 pad-string-left

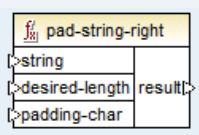
単一の特定文字により左にパッドされた文字列を返します。希望する文字列の長さとパッディング文字は引数として与えられます。



string	入力文字列を指定する。
desired-length	パッディングの後の文字列の希望する長さを定義する。
padding-char	パッディング文字として使用する文字を定義する。

8.8.18.11 pad-string-right

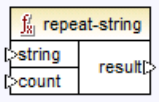
単一の特定文字により右にパッドされた文字列を返します。希望する文字列の長さとパッディング文字は引数として与えられます。



string	入力文字列を指定する。
desired-length	パッディングの後の文字列の希望する長さを定義する。
padding-char	パッディング文字として使用する文字を定義する。

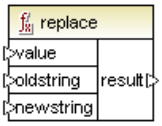
8.8.18.12 repeat-string

引数 *n* 回与えられた文字列を繰り返します。**count** 引数は、文字列の繰り返しの回数を指定します。



8.8.18.13 replace

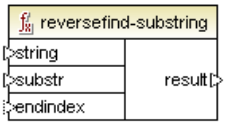
value により得られた入力文字列のうち、**oldstring** の文字列に一致する各文字列が**newstring** の文字列に置き換えられます。



例に関しては、次を参照してください[特殊文字の置き換え]。

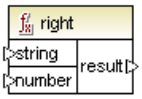
8.8.18.14 reversefind-substring

string 文字列内で**substr** が最初に出現する位置を返します。**endindex** にて検索が開始される場所が指定され、右から左へ検索が行われます。最初の文字の位置は 1 となります。substring が見つからなかった場合、0 が返されます。



8.8.18.15 right

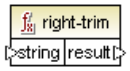
string 内にある**number** により指定された数の文字（列）が最後から返されます。



例：string="This is a sentence" でnumber=5 の場合、"tence" が返されます。

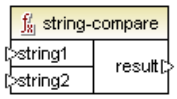
8.8.18.16 right-trim

与えられた文字列の終端にある連続した空白スペースを削除して返します。



8.8.18.17 string-compare

string1 と**string2** の文字列を比較した結果を返します（大文字と小文字は区別されます）。string1 とstring2 が等しい場合、0が返されます。

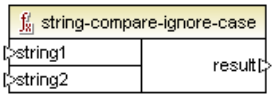


string1 がstring2 よりも小さい場合は、0より小さな値が返されます。

string1 がstring2 よりも大きい場合は、0より大きな値が返されます。

8.8.18.18 string-compare-ignore-case

string とstring2 の文字列を比較した結果を返します（大文字と小文字の違いは無視されます）。string1 とstring2 が等しい場合、0が返されます。

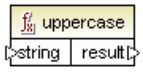


string1 がstring2 よりも小さい場合は、0より小さな値が返されます。

string1 がstring2 よりも大きい場合は、0より大きな値が返されます。

8.8.18.19 uppercase

入力から得られた文字列を大文字に変換して返します。Unicode の文字の場合、（Unicode コンソーシアムにより定義された）対応する大文字が使用されます。

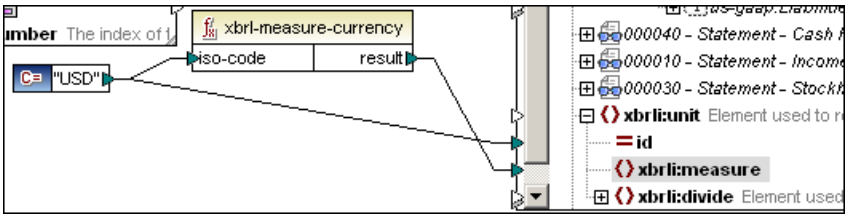


8.8.19 xbrl

XBRL ライブラリには、XBRL インスタンス/タクソノミファイルに必要なQName フォーマットへ、データを変換するための関数が用意されています。

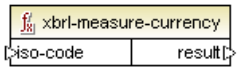
xbrl:unit 要素のxbrl:ID とxbrl:measure 子要素はXBRL インスタンスファイルで必須の要素となっており、妥当な

ファイルを作成するためにはマッピングを行う必要があります。



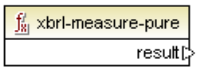
8.8.19.1 xbrl-measure-currency

ISO 4217 通貨コードから、monetaryItemType のアイテムに対して QName を返します。



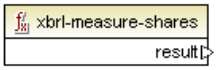
8.8.19.2 xbrl-measure-pure

レート、パーセンテージ、または比率に対して QName が返されます。



8.8.19.3 xbrl-measure-shares

shareItemType のアイテムに対して QName が返されます。



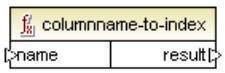
8.8.20 xlsx

XLSX ライブラリには Excel の日付/時間フォーマットとのデータ変換を行うための関数が用意されています。

C# のコードを生成する際には、これらの関数が .NET 3.0 以降を必要としている Excel コンポーネントに依存しており、関数を利用するには「ツール| オプション| 生成 | C#」設定にて、対応する Visual Studio 2005 / 2008 / 2010 を選択する必要があることに注意してください。

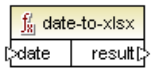
8.8.20.1 columnname-to-index

与えられた名前に対するカラムのインデックスが返されます。"A" のカラムが 1 に対応します。



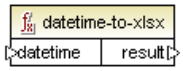
8.8.20.2 date-to-xlsx

与えられたdate / time / datetime の値をExcel の書式に変換して返します。



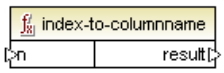
8.8.20.3 datetime-to-xslx

与えられたdatetime 値をExcel のdatetime 書式に変換して返します。



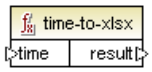
8.8.20.4 index-to-columnname

カラム 1が "A" に対応するカラムの名前を返します。



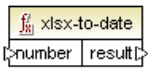
8.8.20.5 time-to-xlsx

与えられた時刻 (time) の値をExcel の時刻書式に変換して返します。



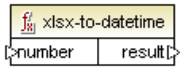
8.8.20.6 xlsx-to-date

Excel の日付フィールドからdate 型の値を返します。



8.8.20.7 xlsx-to-datetime

Excel の日時フィールドからdateTime 型の値を返します。



8.8.20.8 xlsx-to-time

Excel の時刻フィールドからtime 型の値を返します。



8.8.21 xpath2 | accessors

XPath2 関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。

8.8.21.1 base-uri

base-uri 関数はパラメーターにノードを受け取り、そのノードが含まれるXML リソースのURI を返します。出力の型はxs:string になります。入力ノードが与えられていない場合、MapForce によりエラーが表示されます。

8.8.21.2 node-name

node-name 関数は入力パラメーターとしてノードを受け取り、そのQName を返します。QName が文字列として表示され、ノードにプレフィックスがある場合はprefix:localname 形式の入力を受け取り、ノードにプレフィックスが無い場合は、localname 形式の入力が受け取られます。ノードの名前空間 URI を取得するには、(name-に関連したfunctions 以下にある) namespace-URI-from-QName 関数を使用してください。

8.8.21.3 string

`string` 関数は、`xs:string` のように作動し、入力された値を`xs:string` 型に変換します。

入力が (例えば`xs:decimal` の) 原子型である場合、原子型の値が`xs:string` 型の値に変換されます。入力パラメーターがノードの場合、ノードの文字列値が抽出されます (ノードの文字列値とはノードの子孫ノードから得られた値を連結したものになります)。

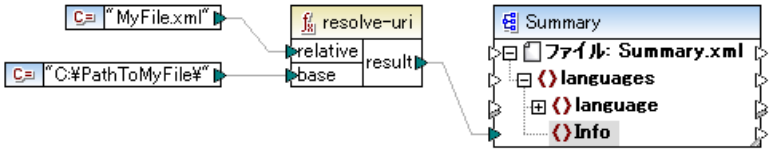
8.8.22 xpath2 | anyURI functions

XPath2 関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。

8.8.22.1 resolve-uri

revolve-uri 関数は最初のパラメーター (elative) としてURI (データ型：xs:string) を受け取り、2番目のパラメーター (base) から得られたベースURI (データ型：xs:string) に対してURI を解決します。

結果 (データ型`xs:string`) は結合されたURI です。この方法で、相対 URI (最初の引数) は、ベースURI に対して解決されることで、絶対 URI に変換されます。



上のスクリーンショットでは、最初のパラメーターにより相対 URI が与えられ、 2番目のパラメーターによりベースURI が与えられます。解決されたURI はベースURI と相対 URI が連結され、C:\PathToMyFile\MyFile.xml が得られます。

メモ: パラメーターは両方ともxs:string 型で、両方の入力を文字列として扱うことで処理は行われます。そのため、解決されたURI のリソースが実際に存在するかどうかはチェックされません。 2番目のパラメーターが与えられていない場合、MapForce によりエラーが表示されます。

8.8.23 xpath2 | boolean functions

XPath2 関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。
boolean 関数として収められているtrue ならびにfalse 関数はパラメーターを受け取ること無く、boolean の定数値であるtrue とfalse をそれぞれ返します。定数のboolean 値が必要な場所で使用することができます。

8.8.23.1 false

boolean 値の "false" を挿入します。

8.8.23.2 true

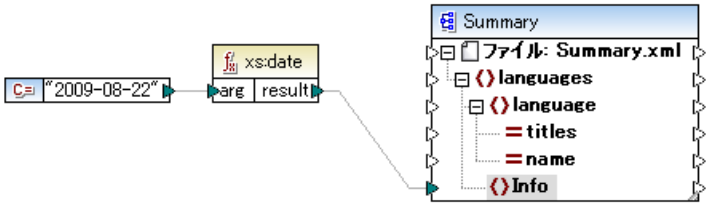
boolean 値の "true" を挿入します。

8.8.24 xpath2 | constructors

XPath2 関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。

XPath 2.0 関数ライブラリのconstructors 以下にある関数を使用することで、入力テキストから特定のデータ型を構築することができます。通常、入力されたテキストは構築されるデータ型に対応した書式で記述されている必要があり、そうでない場合は変換に失敗します。

例えば、xs:date データ型を構築したい場合、constructors 以下にあるxs:date 関数を使用します。入力テキストはxs:date データ型の書式であるYYYY-MM-DD の形式で記述されている必要があります (以下のスクリーンショットを参照ください)。



上のスクリーンショットでは、文字列定数 (2009-08-22)が関数への入力パラメーターとして使用されています。入力はソースドキュメント内のノードからも得ることができます。

xs:date 関数により、xs:string 型で与えられた入力テキストを値とする (関数により指定されている) xs:date データ型が出力されます。

関数ボックスの入力パラメーター上にマウスを移動させると、パラメーターのデータ型がポップアップで表示されます。

8.8.25 xpath2 | context functions

XPath2 関数は XSLT2 または XQuery 言語が選択されている状態で利用することができます。

Context 関数ライブラリには現在の日付や時刻、プロセッサーにより使用されているデフォルトの照合、現在のシーケンスの大きさや現在のノードの位置といった情報を得るための関数が用意されています。

8.8.25.1 current-date

システムクロックから得られた現在の日付 (`xs:date`) を返します。

8.8.25.2 current-dateTime

システムクロックから得られた現在の日時 (`xs:dateTime`) を返します。

8.8.25.3 current-time

システムクロックから得られた現在の時刻 (`xs:time`) を返します。

8.8.25.4 default-collation

default-collation 関数は入力パラメーターを取らず、デフォルトの照合、つまり指定することができる関数に対して特定の照合が指定されなかった場合に使用される照合を返します。

Altova XSLT 2.0 エンジンでは Unicode のコードポイント照合だけがサポートされます。fn:max や fn:min 関数といった比較機能は、この照合をベースに行われます。

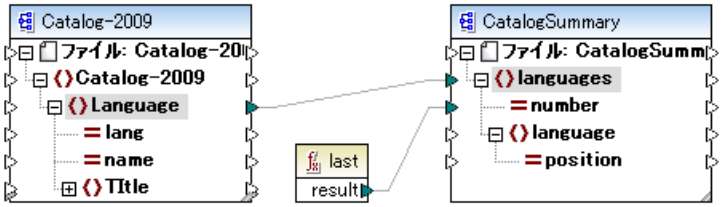
8.8.25.5 implicit-timezone

評価コンテキストからの "implicit timezone" プロパティの値を返します。

8.8.25.6 last

last ならびに position 関数は入力パラメーターを取りません。last 関数により、コンテキストノードセット内の最後にあるノードの場所が返されます。position 関数は、ノードセット内で現在処理されているノードの位置を返します。

関数が接続されているノードのコンテキストノードセットが、関数が適用されるノードセットとなります。以下にあるスクリーンショットでは、language 要素が last ならびに position 関数のコンテキストノードセットとなります。



上の例では、last 関数によりコンテキストノードセット (Language 要素のノードセット) における最後のノード位置が、number 属性の値へ返されます。この値はそのままノードセット内にあるノードの数となり、ノードセットの大きさを表す値にもなります。

position 関数により、処理されているLanguage ノードの位置が返されます。各 Language 要素ノードに対して、Language 要素のノードセット内における位置がlanguage/@position 属性ノードへ出力されます。

core 関数にある**position** ならびに**count** 関数の使用が推奨されます。

8.8.26 xpath2 | durations, date and time functions

XPath2 関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。

XPath 2 duration, date, and time functions 内にある関数を使用することで、タイムゾーンの日時を調整したり、date-time データから特定のコンポーネントを抽出、複数ある日時の値から期間を取得することができます。

'Adjust-to-Timezone' 関数

これら関数はそれぞれ date、time、または dateTime データ型の値を最初の入力パラメーターとして受け取り、2番目のパラメーターから得られた値により、タイムゾーンの削除や編集を行います。

最初のパラメーターにタイムゾーン情報が含まれていない (例えば 2009-01 という日付や 14:00:00 という時刻) 場合、adjust-to-timezone 関数を使った以下のような場合が考えられます：

- (2番目の入力パラメーターである) timezone パラメーターが与えられている場合、2番目のパラメーターから得られた値がタイムゾーンとして使用されます。2番目のパラメーターから得られたタイムゾーンが追加されます。
- timezone パラメーターが与えられていない場合、システムのタイムゾーンである暗黙的なタイムゾーンが結果に含まれます。システムのタイムゾーンが含まれます。
- timezone パラメーターが空の場合、結果にタイムゾーンは含まれません。

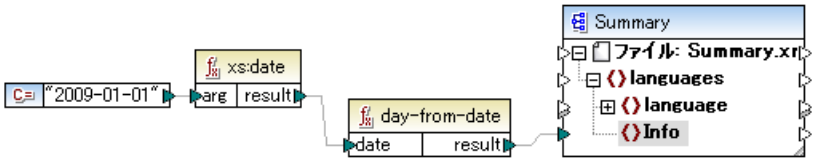
最初のパラメーターにタイムゾーン情報 (例えば 2009-01-01+01:00 という日付や 14:00:00+01:00 という時刻) が含まれている場合、adjust-to-timezone 関数を使った以下のような場合が考えられます：

- (2番目の入力パラメーターである) timezone パラメーターが与えられている場合、2番目のパラメーターから得られた値がタイムゾーンとして使用されます。オリジナルのタイムゾーンが、2番目のパラメーターから得られたタイムゾーンに置き換えられます。
- timezone パラメーターが与えられていない場合、システムのタイムゾーンである暗黙的なタイムゾーンが結果に含まれます。オリジナルのタイムゾーンがシステムのタイムゾーンに置き換えられます。
- timezone パラメーターが空の場合、結果にタイムゾーンは含まれません。

'From' 関数

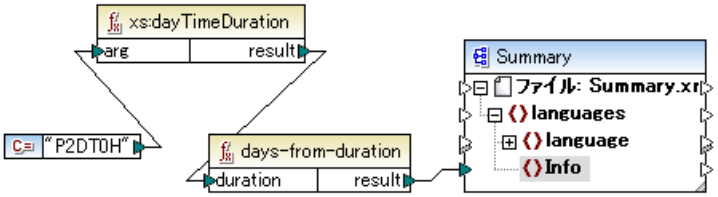
これらfrom 関数は、(i) 日付や時刻のデータ または (ii) 期間を表すデータから特定のコンポーネントを抽出するのに使用されます。xs:decimal データ型の結果が返されます。

日付または時刻のデータから、コンポーネントを抽出する例として、以下のスクリーンショットにあるようなマッピングを使用します。



入力パラメーターにはxs:date 型の日付 (2009-01-01)が与えられています。day-from-date 関数により、xs:decimal データ型の日付コンポーネントが抽出されます。

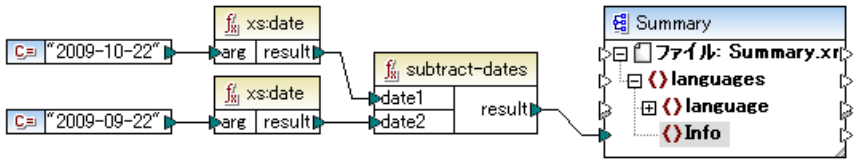
期間型のデータから時刻コンポーネントを抽出する場合、 年と月の抽出するには) 期間をxs:yearMonthDuration または (日、時、分、秒の抽出には) xs:dayTimeDuration として指定する必要があります。xs:decimal 型の結果が得られます。P2DT0H というxs:dayTimeDuration 型の入力をdays-from-duration 関数へ与えているマッピング例を以下のスクリーンショットに示します。結果はxs:decimal 型の 2 となります。



'Subtract' 関数

用意されている 3個のsubtraction 関数により、時間に関する値を他の値から差し引き、期間の値を得ることができます。subtract-dates、subtract-times、そしてsubtract-dateTimes という関数が用意されています。

subtract-dates 関数を使用することで、 2つの日付を差し引いた期間 (2009-10-22 - 2009-09-22) を求める例を、以下のスクリーンショットにて示します。xs:dayTimeDuration 型のP30D が結果として得られます。

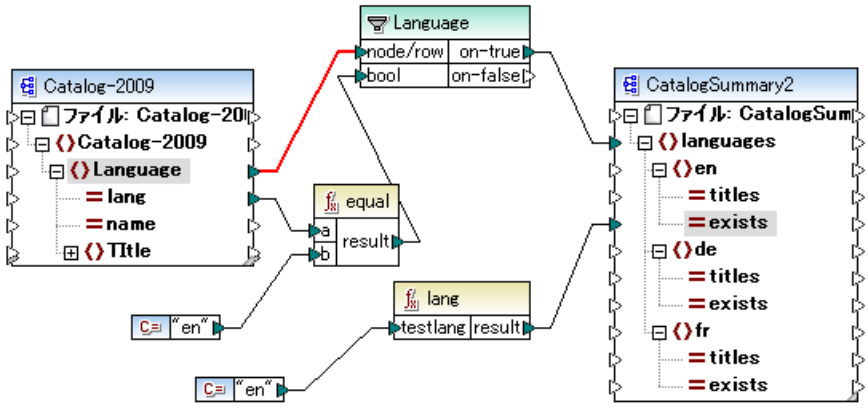


8.8.27 xpath2 | node functions

以下のnode 関数が用意されています：

lang

lang 関数は (例えばen という) 言語コードが書かれた文字列を受け取ります。コンテキストノードにxml:lang 属性が存在しており、属性値が関数のパラメーターで指定された値にマッチするかがチェックされ、true または false が返されます。



上のスクリーンショットについて、以下の点に注意してください：

1. ソーススキーマ内にあるLanguage 要素以下には、xml:lang 属性があります。
2. Language 要素がフィルタリングされ、en という値を持った xml:lang 型の要素だけが処理されます（フィルタリングを行うための条件は equal 関数により与えられます）。
3. Language ノードがコンテキストノードとなり、出力ドキュメントではen 要素が作成されます。
4. lang 関数の出力（true または false）が、出力コンポーネントのen/@exists 属性へ渡されます。関数のパラメーターにはen という定数の文字列が与えられます。lang 関数により、コンテキストノード（Language 要素）に e n という（関数のパラメーターから得られた）値を持ったxml:lang 属性があるかどうかがチェックされます。もしそのような属性値が存在する場合、true が返され、それ以外の場合はfalse が返されます。

local-name, name, namespace-uri

local-name、name、namespace-uri 関数により、それぞれローカル名、名前、そして入力ノードの名前空間 URI が返されます。例えば、altova:Products というノードがある場合、Products がローカル名になり、名前が altova:Products、そして名前空間 URI は、altova: というプレフィックスに対応した名前空間のURI （例：http://www.altova.com/examples）となります。

これらの関数には、それぞれ 2 つの派生があります：

- 入力パラメーターを持たない場合：コンテキストノードに対して関数が適用されます（コンテキストノードのサンプルについては、上にあるlang 関数の例を参照ください）。
- node 入力パラメーターを持つ場合：与えられたノードに対して関数が適用されます。

これら関数の出力はstring 型の文字列となります。

number

入力文字列を数値へ変換します。boolean 型の値も数値へ変換されます。

number 関数はノードを入力として受け取り、ノードを原子化（コンテンツの抽出）して、その値を数値へ変換し、変換された値を返します。変換することのできる値のデータ型はboolean、string、その他の数値型だけになります。数値型ではない値（例えば数値では無い文字列）が入力値の場合、NaN (Not a Number) という値が返されます。

number 関数には二種類あります：

- 入力パラメーターを持たない場合：コンテキストノードに対して関数が適用されます（コンテキストノードのサンプルについては、上にあるlang 関数の例を参照ください）。
- node 入力パラメーターを持つ場合：与えられたノードに対して関数が適用されます。

8.8.28 xpath2 | numeric functions

以下のnumeric 関数が用意されています：

abs

abs 関数は数値型の値を受け取り、その値の絶対値を返します。例えば、入力パラメーターから -2 または +2 が与えられた場合、2 という値が返されます。

round-half-to-even

round-half-to-even 関数により、最初のパラメーターから与えられた値を（2番目のパラメーターから与えられた）オプションの数値精度で四捨五入します。例えば、最初のパラメーターに 2.141567 という値が与えられており、2番目パラメーターに 3 という数値が与えられている場合、最初のパラメーター値が小数点第 3位で四捨五入され、2.141 が返されます。精度（2番目パラメーター）が指定されていない場合、精度に 0 が指定され、与えられた値は整数（nteger）になります。

関数に éven' という名前が付けられている通り、値がちょうど中間に位置している場合、偶数が採用されます。例えば、round-half-to-even(3.475, 2) は 3.48 という値を返します。

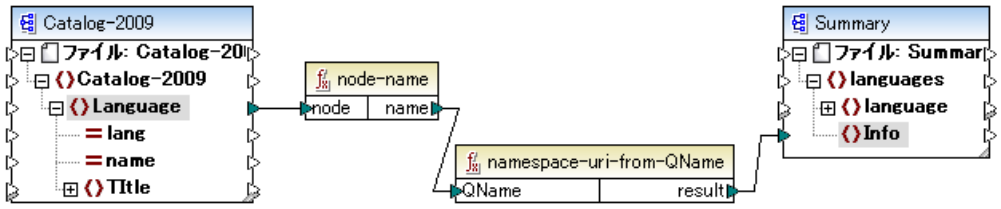
8.8.29 xpath2 | qname- related functions

QName に関連した関数には、local-name-from-QName とnamespace-uri-from-QName という関数が用意されています。

関数は両方とも、文字列形式の）拡張 QName を入力パラメーターとして受け取り、拡張 QName のローカル名と名前空間 URI がそれぞれ返されます。

両方の関数において入力は文字列として処理されるため、これらの関数の入力パラメーターに対して、ノードを直接接続することはできない点に注意してください。

ノードをnode-name 関数へまず接続し、拡張 QName の出力を得ます。node-name 関数の出力から得られた拡張 QName を、これら関数の入力パラメーターへ与えます（以下のスクリーンショットを参照ください）。



これら関数の出力結果はいずれも文字列型になります。

8.8.30 xpath2 | string functions

以下にあるXPath string 関数が用意されています：

compare

compre 関数は、入力パラメーターから与えられた 2つの値がアルファベットとして同一の値かを比較します。string1 の値がアルファベットとしてstring2 のそれよりも小さい場合（例えばA とB という文字列が与えられた場合）、-1 という値が関数から返されます。2つの文字列が同一の場合（例えばA とA）、0 が返されます。string1 の値がstring2 よりも大きい場合（例えばB とA）、+1 が返されます。

この関数の派生型を使用すると、文字列の比較に照合 (collation) を使用することができます。collation パラメーターに値が接続されていない場合、デフォルトの照合である Unicode のコードポイント照合が使用されます。Altova エンジンでは Unicode コードポイント照合だけが利用可能です。

ends-with

ends-with 関数は、string の終端が substring と等しいかをチェックして、等しい場合は true を、そうでない場合は false を返します。

この関数の派生型を使用すると、文字列の比較に照合 (collation) を使用することができます。collation パラメーターに値が接続されていない場合、デフォルトの照合である Unicode のコードポイント照合が使用されます。Altova エンジンでは Unicode コードポイント照合だけが利用可能です。

escape-uri

`escape-uri` 関数は、URI を最初の文字列の引数を入力として取り、文字列に対しての RFC 2396 の URI エスケープ規約に適用します。URI 内で予約済みの文字がエスケープされる場合、第 2 のブール引数 (`escape-reserved`) は `true()` に設定されます。 (例 "+" または "/")。

例 :

`escape-uri("My A+B.doc", true())` would give `My%20A%2B.doc`
`escape-uri("My A+B.doc", false())` would give `My%20A+B.doc`

lower-case

matches 関数により、(input パラメーターにより) 与えられた文字列が、(pattern パラメーターから得られる) 正規表現にマッチするかがチェックされます。正規表現の構文は、XML スキーマの pattern ファセットにて定義する必要があります。文字列が正規表現にマッチすれば true が返され、それ以外の場合は false が返されます。

matches

matches 関数により、(input パラメーターにより) 与えられた文字列が、(pattern パラメーターから得られる) 正規表現にマッチするかがチェックされます。正規表現の構文は、XML スキーマの pattern ファセットにて定義する必要があります。文字列が正規表現にマッチすれば true が返され、それ以外の場合は false が返されます。

オプションの flags パラメーターを関数に与えることもできます。 4 つのフラグ (i、m、s、x) を指定することができます。例えば imx のように、複数のフラグを使用することもできます。フラグが使用されていない場合、 4 つのフラグのデフォルト値が使用されます。

4種類あるフラグの説明を以下に示します :

- `i` 大文字と小文字の違いを無視します。デフォルトでは大文字と小文字は異なるものとして扱われます。
- `m` 複数行モードを使用して、改行文字 (0xa) により複数行に分けられている入力文字列を、1 つの文字列として扱います。メタ文字の ^ と $ が、それぞれ行頭と行末を表します。デフォルトは文字列モードになっており、個々の文字列が ^ から始まり $ で終わります。
- `s` ドット (.) の適用範囲が拡張されます。メタ文字のドット "." はデフォルトで改行文字を除く全ての文字にマッチしますが、このオプションにより、改行文字もマッチするようになります
- `x` 空白スペースを無視します。デフォルトでは、空白スペースは無視されません。

normalize-unicode
normalize-unicode 関数を使用することで、string パラメーターから得られた入力文字列を、normalizationForm パラメーターから得られたルールに従うかたちで正規化します。正規化のフォームとしてNFC、NFD、NFKC、そしてNFKDがサポートされています。

replace
replace 関数では、input パラメーターから得られた文字列に対して、pattern パラメーターで指定された正規表現を適用し、マッチした箇所をreplacement パラメーターの値により置き換えます。

マッチングに使用されるルールについては、上にあるmatches 関数の記述を参照ください。この関数にはオプションとしてflag パラメーターを与えることもできます。flag にて指定できる内容については、matches 関数の記述を参照ください。

starts-with
stars-with 関数は、string の始端がsubstring と等しいかをチェックして、等しい場合はtrue を、そうでない場合はfalse を返します。

この関数の派生型を使用すると、文字列の比較に照合 (collation) を使用することができます。collation パラメーターに値が接続されていない場合、デフォルトの照合であるUnicode のコードポイント照合が使用されます。Altova エンジンではUnicodeコードポイント照合だけが利用可能です。

substring-after
substring-after 関数では、（2番目のパラメーターである）arg2 にて得られた値が（最初のパラメーターである）arg1 から得られた文字列に対してマッチされ、マッチした部分より後ろの文字列が返されます。オプションである 3番目のパラメーターを使用することで、文字列の比較に使用する照合を指定することができます。collation パラメーターに値が接続されていない場合、デフォルトの照合であるUnicode のコードポイント照合が使用されます。Altova エンジンではUnicodeコードポイント照合だけが利用可能です。

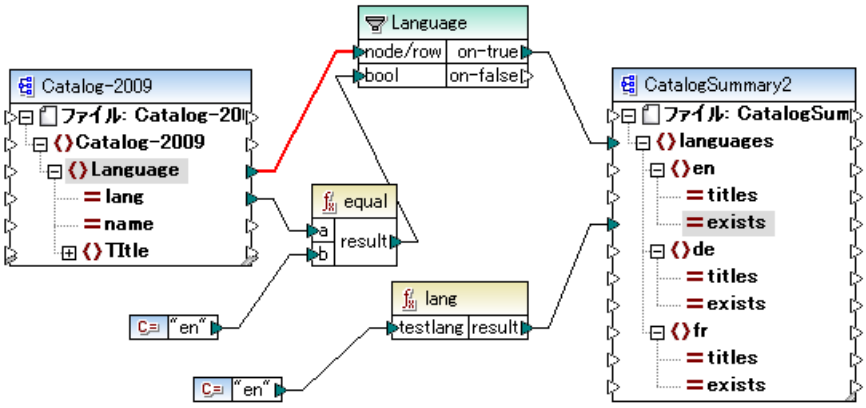
substring-before
substring-before 関数では、（2番目のパラメーターである）arg2 にて得られた値が、（最初のパラメーターである）arg1 から得られた文字列に対してマッチされ、マッチした部分までの文字列が返されます。オプションである 3番目のパラメーターを使用することで、文字列の比較に使用する照合を指定することができます。collation パラメーターに値が接続されていない場合、デフォルトの照合であるUnicode のコードポイント照合が使用されます。Altova エンジンではUnicodeコードポイント照合だけが利用可能です。

upper-case
upper-case 関数は、入力パラメーターから受け取った文字列の中にある小文字を、大文字に変換します。

8.8.31 xslt | xpath functions

xpath functions ライブラリに収められている関数は、XPath 1.0 ノードセットの関数です。各関数は、ノードやノードセットをコンテキストとして受け取り、そのノードやノードセットに関する情報を返します。これらの関数は、通常以下のような要素を備えています：

- コンテキストノード：以下のスクリーンショットでは、ソーススキーマにあるLanguage 要素がlang 関数のコンテキストノードとして与えられています。
- 入力パラメーター：以下のスクリーンショットでは、文字列定数のen がlang 関数の入力パラメーターとして与えられています。last ならびにposition 関数は入力パラメーターを取りません。



lang

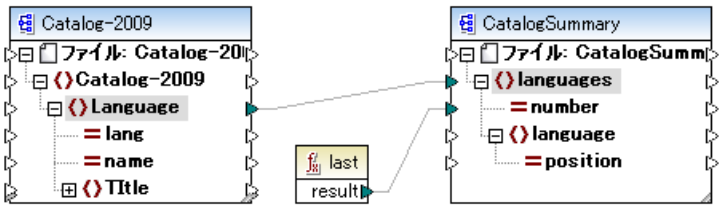
lang 関数は (例えば en という) 言語コードが書かれた文字列を受け取ります。コンテキストノードに xml:lang 属性が存在しており、属性値が関数のパラメーターで指定された値にマッチするかがチェックされ、true または false が返されます。上のスクリーンショットにおいて、以下の点に注目してください:

1. ソーススキーマにある Language 要素には xml:lang 属性が含まれています。
2. `Language` ノードはフィルタリングされており、`en` という値の `xml:lang` 属性を持った `Language` ノードだけが処理されます (フィルタリングの条件が `equal` 関数により与えられます)。
3. Language ノードがコンテキストノードとなり、出力ドキュメントにて en 要素が作成されます。
4. lang 関数の出力 (true または false) が出力ファイルの en/@exists 属性ノードへ渡されます。定数の en が関数のパラメーターへ渡されています。lang 関数はコンテキストノード (Language 要素) に xml:lang 属性があり、その値が (関数のパラメーターにより与えられた) en であるかチェックして、そうである場合は true を、そうでない場合は false を返します。

last, position

last ならびに position 関数は入力パラメーターを受け取りません。last 関数では、コンテキストノードセット内にある最後のノードの位置が返されます。position 関数により、処理中のノードセット内にある現在のノード位置が返されます。

関数が接続されているノードにあるコンテキストノードセットは、関数が適用されるノードセットとなります。以下のスクリーンショットにおいて、last 関数のコンテキストノードセットは Language 要素になります。



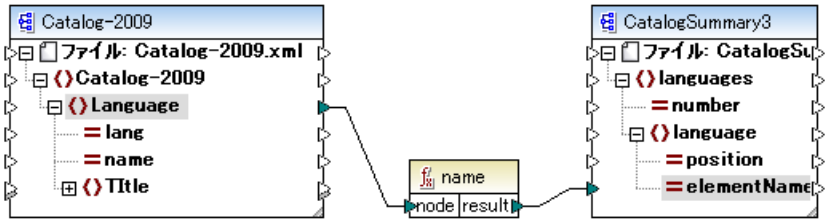
上の例では last 関数により、コンテキストノードセット (Language 要素のノードセット) にある最後のノードの位置が、number 属性の値へ返されます。この関数はノードセット内にあるノードの数を返すため、ノードセットの大きさを示す値が返されます。

position 関数は、Language ノードにおいて現在処理されているノードの位置を返します。Language 要素のノードセット内にて処理されているノードの位置が language/@position 属性ノードへ返されます。

name, local-name, namespace-uri

これらの関数は同じように使用され、それぞれ入力ノードの名前、ローカル名、名前空間 URI を返します。以下にあるスクリーンショットにて、これら関数の使用方法を確認することができます。コンテキストノードが指定されていない点に注目してください。

name 関数は、Language ノードの名前をlanguage/@elementname 属性に対して返しています。これら関数のパラメーターが、単一のノードではなく、ノードセットの場合、そのノードセットにある最初のノードの名前（またはローカル名や名前空間 URI）が返されます。



name 関数は、ノードの QName を返します。local-name 関数により、ノード QName のローカル名部分が返されます。例えば、ノードの QName が altova:MyNode となっている場合、MyNode がローカル名になります。

名前空間 URI とは、ノードが属している名前空間の URI のことです。例えば以下のような方法で、altova: というプレフィックスを、名前空間 URI へマッピングすることができます：xmlns:altova="http://www.altova.com/namespaces"。

メモ： その他の XPath 1.0 関数を core 関数 ライブラリ以下で確認することもできます。

8.8.32 xslt | xslt functions

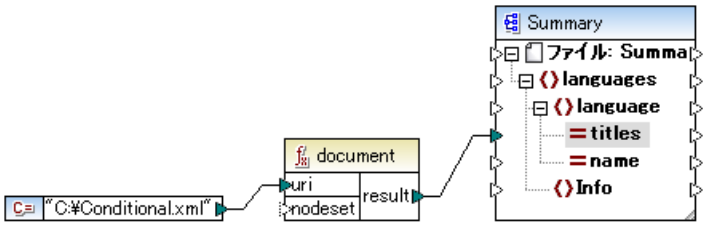
xslt functions ライブラリ内には、以下に記される XSLT 1.0 関数が収められています。

8.8.32.1 currrent

current 関数は入力パラメーターを取らず、現在のノードを返します。

8.8.32.2 document

document 関数により、uri パラメーターに接続した外部 XML ドキュメントを開くことができます（以下のスクリーンショットを参照）。uri パラメーターで与えられた URI が相対 URI の場合、URI を解決するためのベース URI を、オプションの nodeset パラメーターから指定することができます。関数の結果は、出力ドキュメント内にあるノードへ返されます。

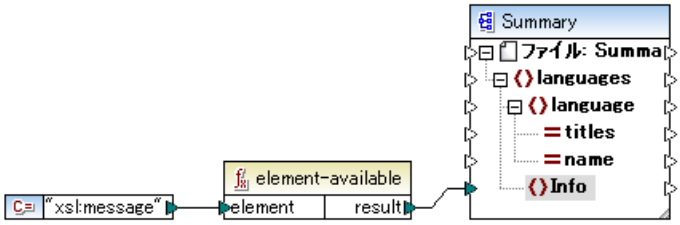


uri パラメーターに与えられる文字列は絶対ファイルパスでなければならない点に注意してください。

8.8.32.3 element-available

element-available 関数により、入力された要素が XSLT の処理系によりサポートされているかチェックすることができます。

パラメーターはQName として評価されます。そのため、XSLT 要素にはxsl: プレフィックスを、XML スキーマ要素にはxs: プレフィックスを付与する必要があります。マッピングで生成されるXSLT 内で使用される名前空間にこれらのプレフィックスが対応しています。



関数からはブール値が返されます。

8.8.32.4 function-available

function-available 関数はelement-available 関数と同様に、パラメーターにより与えられた名前の関数が、XSLT の処理系によりサポートされているかチェックするために使用することができます。

入力文字列はQName として評価されます。関数からはブール値が返されます。

8.8.32.5 generate-id

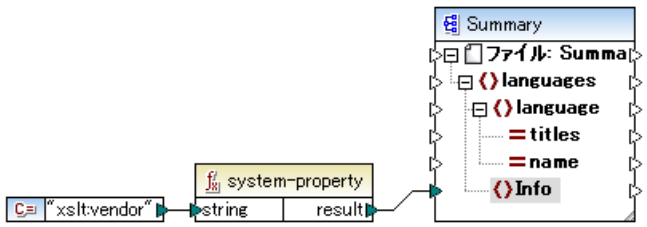
generate-id 関数を使用することで、オプションの入力パラメーターにより与えられたノードセット内における最初のノードを識別するユニークな文字列が生成されます。

入力パラメーターが与えられない場合、コンテキストノード上のID が生成されます。出力ドキュメントにある任意のノードに対して出力結果を接続することができます。

8.8.32.6 system-property

system-property 関数からはXSLT 処理系（システム）のプロパティが返されます。XSLT の処理系にはXSLT 名前空間として書かれたシステムプロパティを指定する必要があります。名前空間はxsl:version、xsl:vendor、そしてxsl:vendor-url となります。

入力された文字列はQName として評価されるため、XSLT スタイルシート内でXSLT 名前空間に関連付けられているxsl:prefix を使用する必要があります。



8.8.32.7 unparsed-enity-uri

DTD を使用している場合、パースされていないエンティティを宣言することができます。パースされていないエンティティ（例えば画像イメージ）は、そのエンティティがある場所を指し示すURI を持つことになります。

関数への入力文字列はDTD 内部にて宣言された（パースされていない）エンティティーにマッチする必要があります。パースされていないエンティティのURI が関数により返され、出力ドキュメント内にあるノード（例えばhref ノード）へ接続することができます。

Chapter 9

Web サービスの実装

9 Web サービスの実装

MapForce Enterprise エディションを使用して、既存のWeb サービス記述言語 (WSDL) ファイルをベースにした SOAP Web サービスを実装するJava またはC# プログラムコードを生成することができます。With MapForce を使用して、以下のようにデータをWSDL オペレーションにマップすることができます :

- フラットファイル、XML、XBRL、EDI、Microsoft Excel、とデータベースを含むMapForce によりサポートされている 全てのデータベースへのオWSDL ペレーションからの入力。
- MapForce によりサポートされるデータソースからWSDL オペレーションへの出力。

MapForce はWSDL 1.1 とWSDL 2.0 をサポートします 各言語のための特定のサポートについての詳細が存在することに注意してください。 WSDL サポート情報も参照)。

必要条件

MapForce でWeb サービスを作成するには、Web サービスのWSDL ファイルが必要です。WSDL ファイルをデザインし、XMLSpy でSOAP リクエストをテストすることも可能です。更に、 Web サービスサーバーを実行するためにプラットフォーム特定のソフトウェアが必要になる場合があることに注意してください。例 :

Java

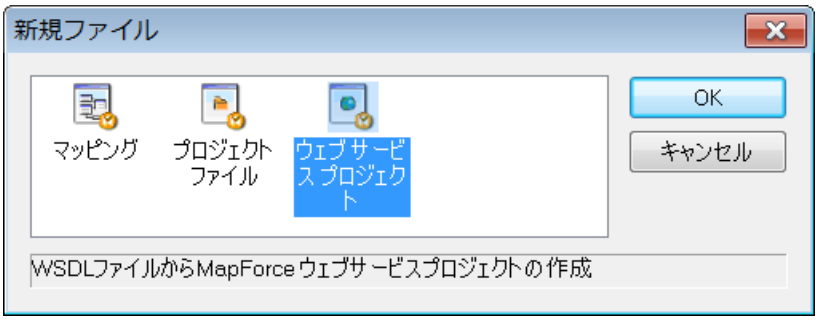
- Java Software Development Kit 1.6 または以降
- Apache Tomcat: http://tomcat.apache.org
- Apache Axis2: http://ws.apache.org/axis2/、Tomcat 内のSOAP フレームワーク
- Apache Ant: http://ant.apache.org/

C#

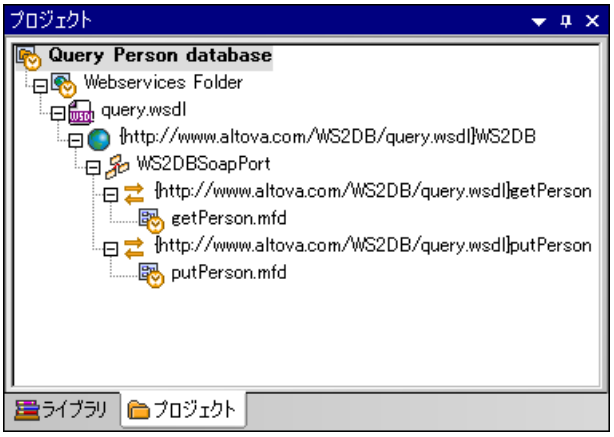
- Microsoft Visual Studio 2005 または以降
- Microsoft Internet Information Services (IIS) verson 5.0 または以降 .

作動のしくみ

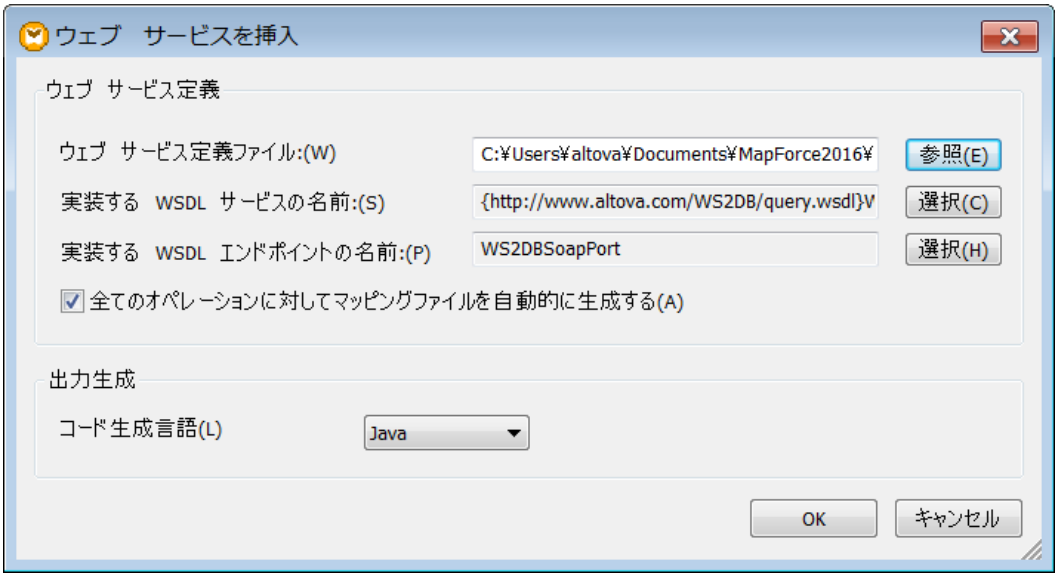
WSDL ファイルの準備ができると、新規 MapForce Web サービス プロジェクトを開始します。



MapForce Web サービス プロジェクトには、定義済みの構造があり、素早く特定のWSDL オペレーション または Web サービスにアクセスすることができます。



MapForce Web サービス プロジェクトを開始する際、オプションで MapForce に WSDL ファイル内のすべての WSDL オペレーションのために自動的にマッピングファイルを生成することを命令することができます。



代わりに、 MapForce Web サービス プロジェクトに含まれるように、WSDL サービスと WSDL エンドポイントを個別に選択することもできます。

MapForce グラフィカルユーザーインターフェイス内で、マッピングに対応する WSDL オペレーションと、オペレーションの入力と出力パラメーターがコンポーネントとしてマッピング内に表示されます。

入力または WSDL オペレーションの出力を含むマッピングと作業する場合、他のマッピングのために行うように実行の結果をプレビューすることができます。これを行うには、WSDL コンポーネント内のサンプル入力 SOAP インスタンスを設定します。BUILT-IN を変換言語として選択し、 **出力** タブをクリックします。

Web サービス プロジェクトが MapForce 内で準備されると、(テスト目的で)個別のマッピングのために、または、プロジェクト全体のために C# または Java コードを生成することができます。後者の場合、MapForce は、選択された言語 (C# または Java) で、Web サービス サーバーを作成することができます。

最後に、生成された C# または Java コードをコンパイルして、(MapForce の外部の) カスタム Web サーバーにデプロイすることができます。

生成されたC# またはJava コードのWeb サービスサーバーへのデプロイは、使用中の第三者ツールと環境に影響されるものであり、このドキュメントでは、例としてのみ扱われています。第三者のソフトウェア、およびウェブサーバーのインストールと構成の詳細に関しては、対応するドキュメントを参照してください。

(C) 2017 Altova GmbH

9.1 WSDL サポート情報

MapForce 内でのWSDL へのサポートの詳細をまとめたテーブルです

WSDL サポート	バージョン 1.1 http:/ / www.w3.org/ TR/ wsdl のW3C ノート バージョン 2.0http://www.w3.org/TR/wsdl20/ のW3C 勧告
WSDL 型システム	XML スキーマ 2001
SOAP サポート	バージョン 1.1:http:/ / www.w3.org/ TR/ 2000/ NOTE- SOAP-20000508/ バージョン 1.2:http://www.w3.org/TR/soap12-part0/
プロトコル	SOAP over HTTP（HTTP POST、HTTP GET プロトコルはサポートされていません）。
C#	SOAPAction は、C# 内の各オペレーションに対して異なるものである必要があります。
バインディング	同名の複数オペレーションは、現在サポートされていません (WSDL 1.1 仕様のWSDL 2.5)。
style/use	• document/literal：サポート • RPC/literal：C# にてサポート • RPC/encoded：限定されたサポート • 1 つのWeb サービス (Java) またはオペレーション (C#) に対して 1 つのstyle/use が現在サポートされています。
SOAP ヘッダー	使用しているプラットフォームに依存
SOAP encodingStyle	"**http://schemas.xmlsoap.org/soap/encoding/**" がsoap:Body 全体に対して指定され、それ以外のエンコーディングスタイルはサポートされません。 encodingStyle 属性はメッセージ内で無視されます（SOAP 1.1 仕様のSOAP 4.1.1)。
参照	• 外部リソースへの参照は現在サポートされていません(SOAP 1.1 仕様のセクション 5.4.1)。 • 参照は独立した要素に対してのみサポートされます。
SOAP-ENC:Array	リニアアクセスがサポートされます。部分アレイと疎な配列は現在サポートされていません。
カスタム SOAP 強化：	未サポート
スキーマ内にあるデフォルトまたな固定値：	未サポート
SOAP メッセージ以外の検証	検証無しで、フレームワークへ受け渡し。
名前空間	非名前空間のエントリーはWSDL にて不正となるため、サポートされていません（WSDL ならびにXML 1.0）

WSDL 1.1

ポートタイプ

<portType> 要素によりWeb サービスのインターフェースが定義されます：

- 処理する**オペレーション**
- 各オペレーションに入力または出力として含まれる**メッセージ**

タイプ

<types> 要素により、Web サービスにより使用されるデータ型が定義されます。MapForce ではWSDL ファイルにて最も広く使用されている種類の、WSDL 内におけるXML スキーマがサポートされています。MapForce ではこれらの要素（データ型）を（メッセージ）コンポーネント内のアイテムとして表示して、その他のアイテム/制約へ直接マッピングを行います。

メッセージ

<message> 要素により、各メッセージのパートとオペレーションの入力ならびに出力パラメーターのデータ要素が定義されます。これらはクライアントとサーバー間でやり取りされるメッセージとなります。メッセージには、input、output、fault の3種類あります。
MapForce では、各メッセージは（例えばgetPersonSoapIn という名前の）コンポーネントとして表示されており、他のアイテムへのマッピングを行うことができます。メッセージは1つ以上のメッセージパーツからなります。

MapForce にてドキュメントとリテラルを混合して使う場合、**message / part 要素**は **type** ではなく名前空間を参照する必要があります。"element" 属性により、スキーマ内で定義されたグローバル要素を参照します（ns2:Vendor）：

```
<message name="processRequest">
      <part name="inputData" element="ns2:Vendor"/>
</message>
```

以下の場合は、スキーマ内部のtype を参照します：

```
<message name="processRequest">
      <part name="inputData" type="ns2:VendorType"/>
</message>
```

オペレーション

オペレーションは、入力ならびに出力パラメーターとしてメッセージを使用します。オペレーションには以下の要素を含めることができます：

- 1 つのInput メッセージ
- 0 、または 1つのOutput メッセージ
- 0 、または 1つのFault メッセージ

Input メッセージはソースコンポーネントとしてのみ使用することができます。出力ならびにFault メッセージはターゲットコンポーネントとしてのみ使用することができます。

WSDL 2.0

WSDL 2.0 は根底部分でWSDL 1.1 と異なっており、主な違いを以下に記します：

- PortTypes がInterfaces に改名されました
- メッセージとパーツがtypes 要素のXML スキーマ型システムにより定義されます
- Ports はエンドポイントに改名されました
- WSDL 2.0 Operation 入力ならびに出力がXML スキーマにより定義されます

WSDL コンポーネントの (MapForce 内における) コンポーネント設定ダイアログボックスにより、WSDL 1.1 Ports と WSDL 2.0 エンドポイントの両方が表示されます。

9.2 WSDL ファイルからWeb サービスプロジェクトを作成

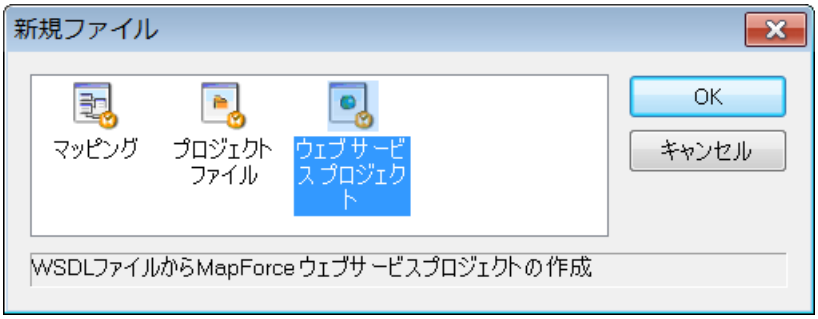
このトピックは、Web サービスを生成するために使用される "Web サービス プロジェクト"の型のプロジェクトの作成方法について説明します。このサンプル内のWeb サービスには以下の目的があります：

- SOAP リクエストを使用することで、MS Access データベースに収められている従業員のセットを検索します。ランタイム時にクライアント側にてクエリの定義 入力が行われ、サーバーへ送信されます。
- データベースレコードをSOAP レスポンスとして取得し、クライントへ送り返されたクエリに一致する従業員を含むXML ドキュメント形式を取得します。

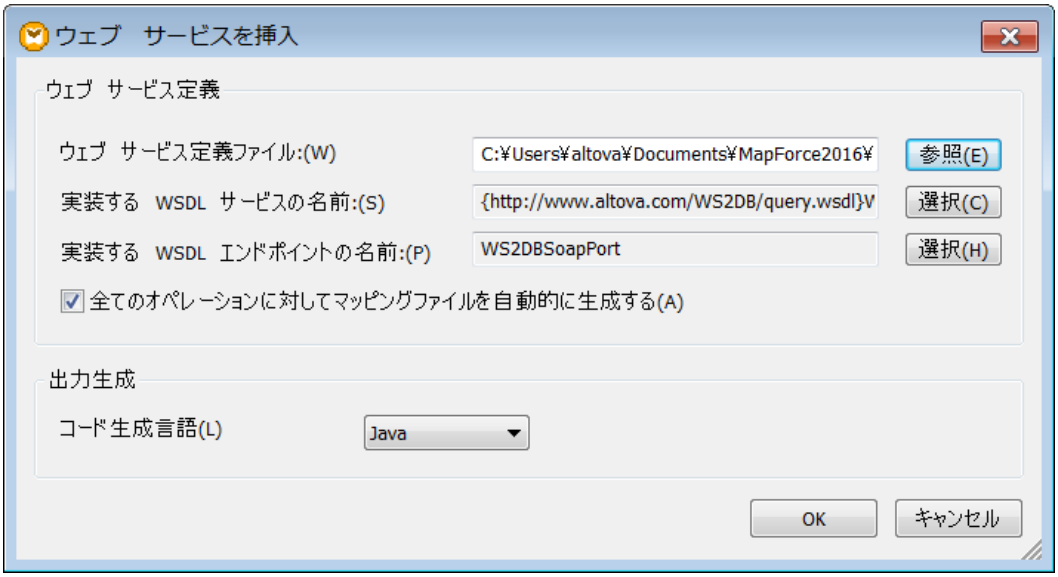
メモ：
Web サービスの生成処理は、ターゲットとなるプログラミング言語に依存せず、Java またはC# にて生成されたWeb サービスは同一のものとなります。Web サービスをコンパイル、デプロイする方法は言語は言語ごとに異なります。

Web サービスプロジェクトを作成する

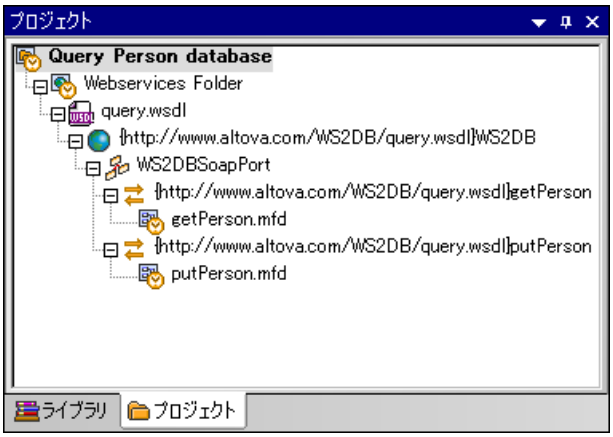
1. メニューオプションの「ファイル| 新規作成」を選択し、Web サービスプロジェクトアイコンをクリックします。



2. フォルダー**<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** に収められている**query.wsdl** を選択します。
3. WSDL ファイルを選択すると、それ以外のフィールドが MapForce により自動的に入力されます。



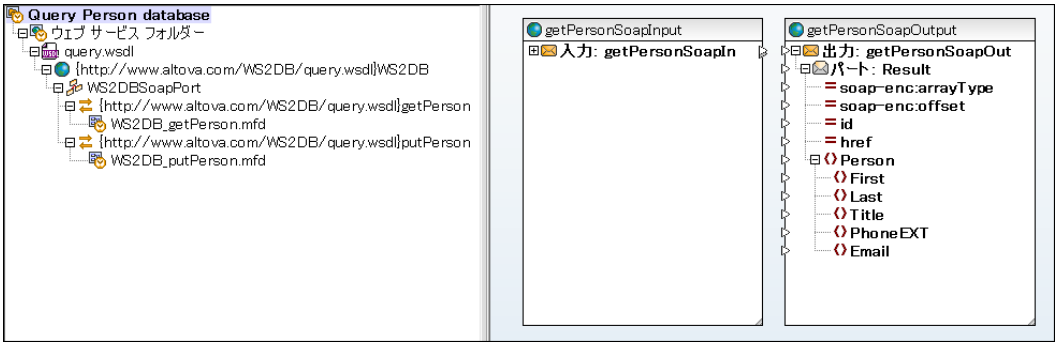
4. **「OK」**をクリックします。
5. WSDL プロジェクトの名前を入力します。**「保存」**をクリックして設定を確定し、WSDL プロジェクトファイルを作成します。



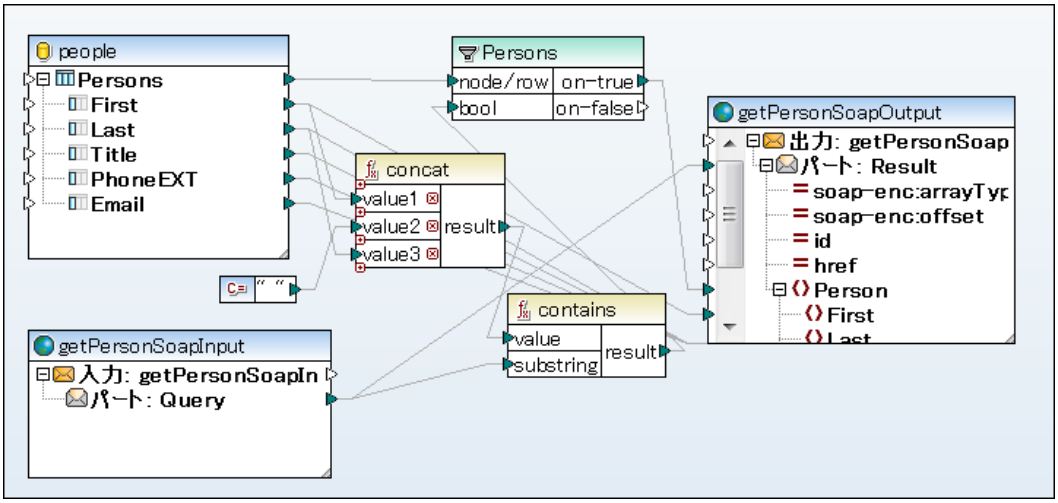
プロジェクトタブには、プロジェクトならびにWSDL 名と WSDL ファイルにて定義されている各オペレーションが表示されます。**getPerson** ならびに**putPerson** という 2 つのオペレーションが存在します。

Web サービスのマッピングを作成する：

1. プロジェクトタブ内にある**WS2DB_getPerson** オペレーションをダブルクリックします。
2. 2 つのWSDL コンポーネントを含む新たなマッピング "**getPerson.mfd**" が作成されます。**getPersonSoapIn** コンポーネントには、Web サービスを通してデータベースへの問い合わせを行うクエリアイテムが含まれます。**getPersonSoapOut** コンポーネントにはWSDL ファイル内のPerson アイテムが含まれています。



3. 「挿入 | データベース」を選択し、ウィザードから**<Documents>\Altova\MapForce2017 \MapForceExamples\Tutorial** フォルダー内のpeople.mdb データベースを挿入します。データベース内のアイテムがgetPersonSoapOut コンポーネントのアイテムにマッチします。
4. `concat` 関数を使用して、従業員のFirst とLast を連結します。
5. **「挿入」**メニューから**「定数」**をクリックします。スペース文字を含む定数を追加します。名と姓の間の空白を定数は提供します。
6. フィルターコンポーネントを追加します。（**「挿入」**メニューから**node/row: 「フィルター」**をクリックします）。
7. コンポーネント間の接続を以下のように作成します。



`getPersonSoapIn` コンポーネントの**part:Query** アイテムは、クエリプレースホルダです（コードが生成、コンパイル、ウェブサーバーにデプロイされると、クエリ文字列がSOAP クライアントに入力される箇所です）。**`contains`** 関数は、クエリ文字列がデータベース内の個人の名前の全体または一部に一致すると、**true** を返します。（"name" は、**`concat`** 関数により返され、名、スペース、姓により構成されます）。一致がある場合、一致する個人の情報が応答メッセージに含まれます。ですから、このサンプルでは、検索される値 "Ma" は、"Martin Rope" と "Mary Morford" の詳細を返します。

これでWeb サービスを作成するためのコードを生成することができます。Web サービスはJava またはC# にて生成することができます。以下にあるセクションでは、Java とC# コードの生成、コンパイル、そしてウェブサーバーにおけるデプロイを行う方法について説明します。

9.3 MapForce にて Java Web サービスを生成する

MapForce では Web サービスを作成するために必要な全てのコードならびにスクリプトが生成されます（コード ジェネレータを参照）。通常のコード生成処理と唯一異なる点は、生成されたコードを Axis2 (Tomcat) サーバーへデプロイする必要があるということです。

Web サービスがリモートコンピューターにて動作している場合：

- リモートにあるマシンの管理者権限が必要になります。
- Axis2 フレームワークを、ローカルのコンピューターへ部分的にインストールする必要があります。

Java コードを生成する：

1. **<Documents>\Altova\MapForce2017\MapForceExamples\Tutorial** フォルダーから前のセクションで使用した **Query Person database.mfp** プロジェクトを開きます。上記のようなプロジェクトの作成に関しては、WSDL ファイルから Web サービス プロジェクトを作成するを参照してください。
2. **プロジェクト** メニューから「プロジェクト | コードの生成 | Java」を選択します。コード生成が完了すると、Altova ジェネリック クラスと Web サービス プロジェクトの実際のクラスなど **com** ディレクトリを含む、複数のフォルダーとファイルがターゲット ディレクトリ内に作成されます。
3. 生成された Java コードをビルドします（MapForce により生成された **build.xml** ファイルを Apache Ant に提供します）。この結果、Axis2 のデプロイに対応した Axis Archive (*.aar) ファイルが作成されます。

生成されたプロジェクトを Eclipse で開くと、'org.apache のインポートが解決されません' のようなエラーが生じる場合があります。この場合、Axis2 ライブラリが Java ビルド パスに追加されていることを確認してください。Axis2 ライブラリを Eclipse 4.4.2 Java ビルド パスに追加するには、以下を行います：

1. パッケージ エクスプローラー内のプロジェクトを右クリックして、**「プロパティ」** を選択します。
2. **「Java ビルド パス」** をクリックします。
3. **「ライブラリ」** タブで **外部 JAR を追加...** をクリックし、Axis2 ライブラリを `<AXIS2_HOME>\lib` フォルダーから追加します。

Web サービスのデプロイ

Web サービスをデプロイするには以下を行います：

- Axis2 の "サービスをアップロード" Web 管理者ページを開き、前のステップで作成された aar ファイルをアップロードします。
- 手動でアップロードします。例：Tomcat サーバーがフォルダー `<TOMCAT_HOME>` にインストールされている場合、手動で aar ファイルを `<TOMCAT_HOME>\webapps\axis2\WEB-INF\services` にコピーすることができます。

デプロイを外す

`<TOMCAT_HOME>\webapps\axis2\WEB-INF\services` フォルダーから *aar ファイルを削除します。

Axis2 の制約

AXIS2 では rpc/encoded に対するサポートが制限されています。MapForce では rpc/encoded Web サービス（SOAP 1.1 ならびに SOAP 1.2）を生成することができます。これにより、オリジナルの WSDL がウェブ サーバーから取得されないという制約があります。

これはつまり、http://127.0.0.1/axis2/services/WS2DB?wsdl により、利用可能な WSDL ファイルが取得**されない**ということになります。

上にある**ドキュメントのリテラル** Web サービスの URL により、正しい WSDL ファイルが与えられますが、それはオリジナルのファイルではありません：コメントは取り除かれ、名前空間が変更されます。それでもオリジナルの WSDL ファイルとは機能的に等価なものとなります。

AXIS2 では **RPC/ encoded** はサポートされていませんが、デプロイされた Java コード（コンパイルされたコード）から WSDL を生成することができ、MapForce により生成されたコードにより **RPC/ encoded** メッセージが処理されます。AXIS2 は移送のためだけに使用されます。

既知の問題：SOAP レスポンスメッセージ内の名前空間

MapForce により生成されたコードは、`javax.xml.transform.TransformerFactory` クラスをインスタンス化します。クラス実装がロードされる場合、`javax.xml.transform.TransformerFactory` システムプロパティから読み込まれる可能性があります。（詳細に関しては、このクラスの Java ドキュメントを参照してください）。

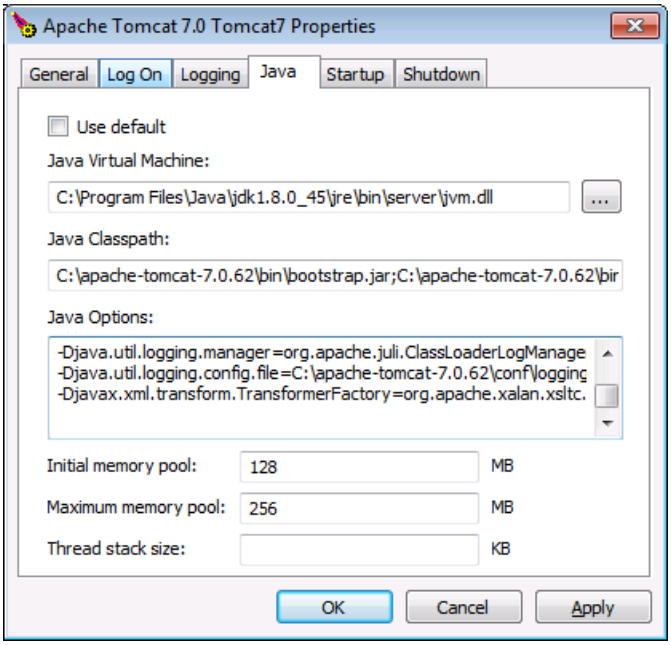
Java 環境内のシステムプロパティ `javax.xml.transform.TransformerFactory` が実装 `org.apache.xalan.transformer.TransformerIdentityImpl` を定義する場合 SOAP レスポンスメッセージが正確ではない名前空間を用いて生成される可能性があります。

この問題を回避するため、`org.apache.xalan.xsltc.trax.TransformerFactoryImpl` 実装を使用してください。これを以下のパラメーターを Java Virtual Machine (JVM仮想マシン)に追加することで行うことができます:

```
-
Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.Transfor
merFactoryImpl
```

パラメーターを JVM に追加する命令は、使用中のオペレーティングシステムとサーバーにより異なります。以下の命令は Windows 7上のサービスアプリケーションとして実行される Apache Tomcat 7.0 構成に適用することができます：

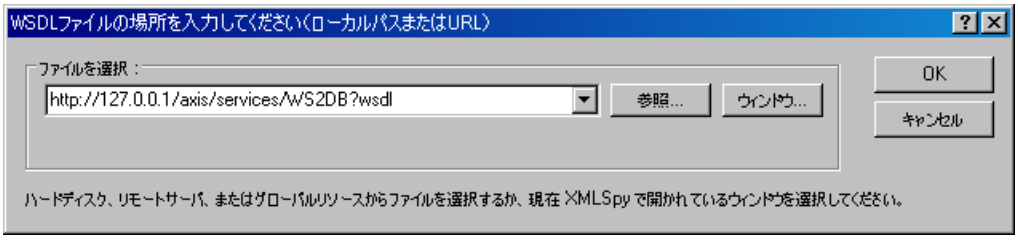
1. **Tomcat7w.exe** を実行します（このファイルは Tomcat 配布ルートフォルダー CATALINA_HOME の **\ bin** サブフォルダー内にあります）。
2. **Java** タブをクリックします。
3. "Java オプション" ボックスの最後に、以下を入力します：-
   `Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.Trans formerFactoryImpl`



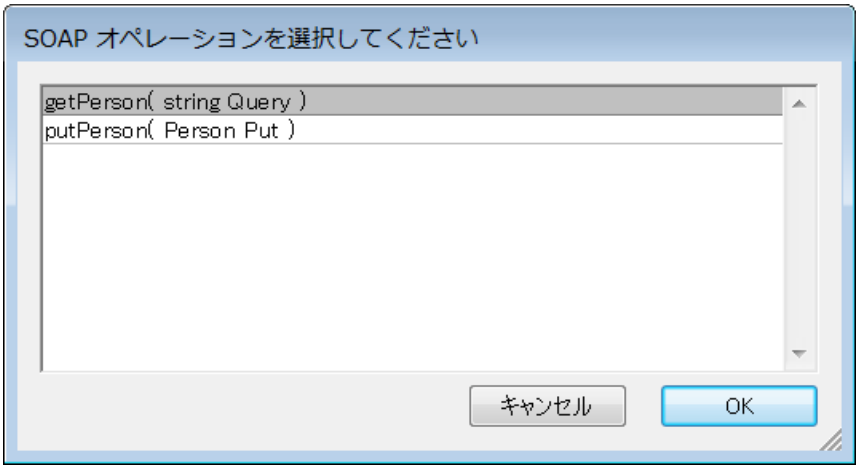
9.3.1 Web サービスを使用する - getPerson オペレーション

Web サービスを使用する：
XMLSpy のようなWSDL クライアントを使用して、SOAP リクエストを作成、送信します。

1. 「**SOAP | 新規 SOAP リクエストを作成**」を選択します。
2. WSDL ファイルの場所を入力し、「OK」をクリックします（例：**http:/ / 127.0.0.1/ axis2/ services/ WS2DB?wsdl**）。



3. 使用するSOAP オペレーション名（この例では "**getPerson(string Query)**"）を選択し、「OK」をクリックしてSOAP リクエストドキュメントを作成します。



<Query xsi:type="xsd.string">**String**</Query> がクエリのプレースホルダになり、データベースにて検索される文字列が含まれます。

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <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>
```

この例では "Ro" という文字列がFirst またはLast 内に含まれている全従業員の名前を検索します。

4. String というテキストを**Ro** に変更します。

```
<m:getPerson xmlns:m="http://www.altova.com/
    <Query xsi:type="xsd:string">Ro</Query>
```

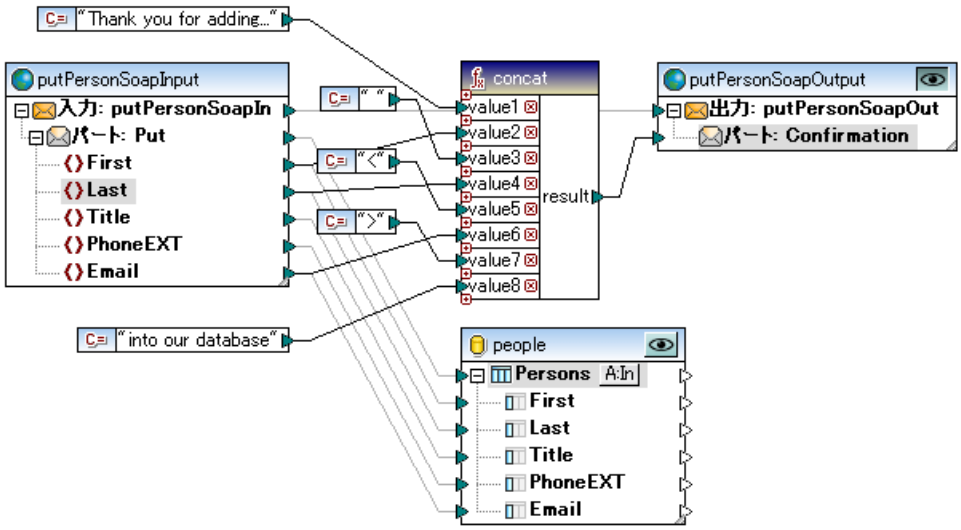
5. メニューオプションから「**SOAP | サーバーにリクエストを送信**」を選択し、サーバーにリクエストを送信します。

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/en
    <soapenv:Body>
        <m:getPersonResponse xmlns:n0="http://www.altova.com/WS2DB.
            <Result soapenc:arrayType="n0:Person[]" xsi:type="soape
                <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</Em
                </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</E
                </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</Tit
                    <PhoneEXT xsi:type="xsd:string">299</PhoneEXT>
                    <Email xsi:type="xsd:string">rodark@strings.com
                </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</Ti
                    <PhoneEXT xsi:type="xsd:string">112</PhoneEXT>
                    <Email xsi:type="xsd:string">rog@strings.com</E
                </Person>
```

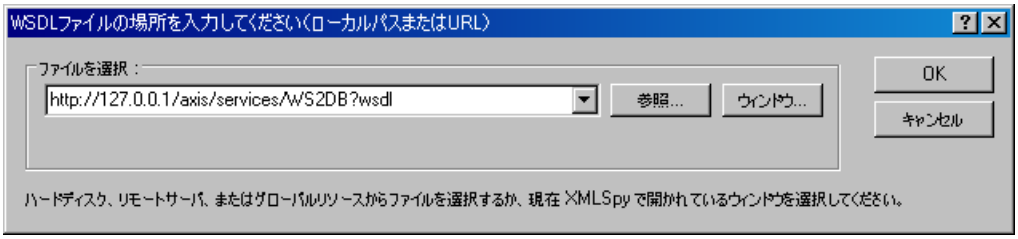
SOAP レスポンスドキュメントがWeb サービスにより返され、Access データベース内にある 4 人分の従業員データが表示されます。

9.3.2 Web サービスを使用する - putPerson オペレーション

- プロジェクトウィンドウ内にある**putPerson** マッピングアイコンをダブルクリックして、マッピングを表示します。**putPerson.mfd** が別のタブで開かれます。



1. XMLSpy のようなWSDL クライアントを使用してSOAP リクエストを作成、送信します。
2. メニューオプションから「**SOAP | 新規 SOAP リクエストを送信**」を選択します。
3. サーバー上にあるWSDL ファイルの場所（例えば**http:/ / 127.0.0.1/ axis2/ services/ WS2DB?wsdl**）を入力し、「**OK**」をクリックします。



4. 使用するSOAP オペレーション名（この例では "**putPerson(Person Put)**"）を選択し、「**OK**」をクリックしてSOAP リクエストドキュメントを作成します。
SOAP リクエストドキュメントには、従業員のプレースホルダが表示されます。

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <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. プレースホルダのテキスト (String) を、実際の従業員データに置き換えます。

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/e
    <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. メニューオプションから「**SOAP | サーバーにリクエストを送信**」を選択して、サーバーへリクエストを送信します。
「SOAP リクエストを送信」コマンドにより、編集されたSOAP リクエストドキュメントがサーバーへ送信されます。

新たな従業員データがデータベースへ追加され、その旨を通知するSOAP レスポンスドキュメントが返信されます。マッピングにより、この通知メッセージの内容 (上のマッピング例では "Thank you for adding XYZ into our database") が定義されます。

```
<?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-instanc
    <soapenv:Body>
        <m:putPersonResponse xmlns:n0="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
&lt;ff@home.com&gt; into our spam database</Confirmation>
        </m:putPersonResponse>
    </soapenv:Body>
</soapenv:Envelope>
```


(C) 2017 Altova GmbH

9.4 MapForce にて C# ウェブサービスを生成

チュートリアルフォルダー内に収められているquery.wsdl ファイルにはコメントアウトされたセクションがあります。<Service name="WS2DB"> というセクションがそれで、ファイルの終端部にあります。このセクションをアンコメントし、その直前にある <Service> セクションをコメントアウトします。この例ではウェブサーバーがローカルのコンピューターで動作していることを前提にしています。

メモ:
AXIS2 (Java) とIIS (C#) のURI スキーマは互いに異なっており、WSDL ファイル内にて変更を行う必要があります。変更は、通常 WSDL ファイルの終端部にある <service> 要素にて行われます。

AXIS2 スタイルのWSDL:

```
<service name="WS2DB">
    <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
        <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
    </port>
</service>
```

IIS スタイルのWSDL:

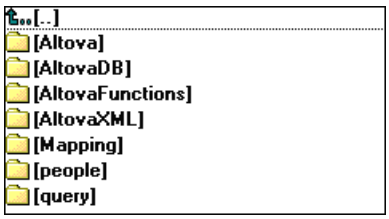
```
<service name="WS2DB">
    <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
        <soap:address location="http://localhost/services/WS2DB.asmx"/>
    </port>
</service>
```

サンプルWSDL ファイルには両方のバージョンが収められており、そのうちどれか 1 つがコメントアウトされています。WSDL サンプルファイルには "soapAction" パラメーターが含まれており、このパラメーターはIIS で必須となりますが、AXIS2 では無視されます。

C# コードを生成する:
前のセクションで使用したQuery Person database.mfp プロジェクトファイルを作成します (又は開きます)。

1. メニューオプションから **「プロジェクト | コードの生成 | C#」**を選択します。
2. 出力ディレクトリ (この場合はjava-dev) を選択し、**「OK」** をクリックして生成を行います。
   "処理が問題なく行われた場合、「C# コードの生成に成功しました」というメッセージが表示されます。生成に失敗した場合、詳細なエラーメッセージが表示されます。

ターゲットディレクトリには、C# コード、プロジェクトファイルならびにソリューションファイルを含む複数のサブディレクトリが自動的に生成されます。



生成されたプロジェクトはVisual Studio 2005/2008/2010 にてコンパイルすることができます。

Web サービスをコンパイルする:
メモ:IIS 7.x とともにVisual Studio 2005 または 2008を使用している場合、「IIS メタベースおよびIIS 6 構成の互換性」("IIS Metabase and IIS 6 configuration compatibility") というWindows の機能をインストールする必要があります。更に、Visual Studio を管理者として起動する必要があります。

以下に記されるステップは、ローカルマシンにウェブサーバーがあることを前提にしています。リモートコンピューターに対して Web サービスのコンパイルを行うには、関連する Microsoft のドキュメンテーションを参照ください（例：http://learn.iis.net/page.aspx/387/using-visual-studio-2008-with-IIS/）。

1. 生成されたソリューションファイルを開きます（例：**...\ output\ Query_Person_database \ Query_Person_database_webservice.sln.**）。
2. メニューオプションから「**ビルド | ソリューションのビルド**」を選択し、Web サービスプロジェクトをコンパイルします。
3. メニューオプションから「**デバッグ | デバッグ開始**」を選択し、アプリケーションを起動します。

64 ビットのオペレーティングシステムを使用していて、（32ビットアプリケーションに対してしか提供されていない）ADO プロバイダーが無いためにサンプルを使用することができない場合、以下の操作を行うことができます。

1. Visual Studio のメニューオプションから「**ビルド | 構成マネージャー**」を選択し、x86 に対する新たなソリューションプラットフォームを作成し、ビルドを行います。
2. Internet Information Services (IIS) マネージャーを開き、作成中のアプリが格納されている「アプリケーションプール」を検索し、右クリックして表示されるコンテキストメニューから「**詳細設定**」を選択し、**32ビットアプリケーションの有効化**にて "True" を選択してください。

Web サービスを使用する：

1. XMLSpy のような WSDL クライアントを使用して、SOAP リクエストを送信します。
2. メニューオプションから「**SOAP | 新規 SOAP リクエストを作成**」を選択します。
3. IIS サーバー上にある WSDL ファイルの場所を入力して、「**OK**」をクリックします。
4. 使用する SOAP オペレーション名（この例では "**getPerson(string Query)**"）を選択し、「**OK**」をクリックして SOAP リクエストドキュメントを作成します。.
   <Query xsi:type="xsd:string">**String**</Query> がクエリのプレースホルダになり、データベースにて検索される文字列が含まれています。

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <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>
```

   ここでは "Ro" という文字列が First または Last 内に含まれている全従業員の名前を検索します。
4. String という文字列を **Ro** に変更します。

```
<m:getPerson xmlns:m="http://www.altova.com/
    <Query xsi:type="xsd:string">Ro</Query>
```

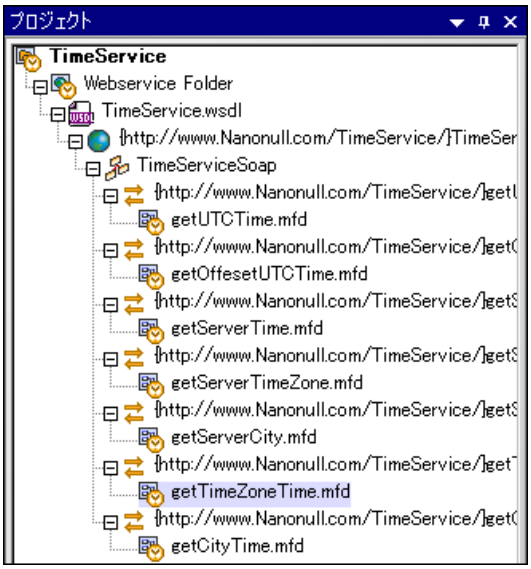
5. メニューオプションから「**SOAP | サーバーにリクエストを送信**」を選択して、サーバーへリクエストを送信します。
   Web サービスにより SOAP レスポンスドキュメントが返され、Access データベースから4人の従業員名が帰ってくることが確認できます。

9.5 Web サービスのフォールト (fault)

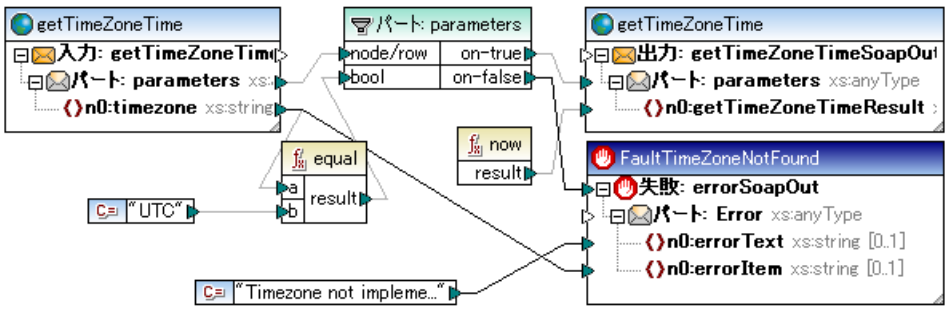
MapForce では WSDL フォルトの定義を行うことができます。WSDL ファイルにはフォルトメッセージを含む "fault" 要素をオペレーションならびにメッセージ属性に追加することができます。fault **要素**を WSDL ファイルに含めることで、Fault コンポーネントをマッピングで扱うことができます。

MapForce では指定された条件が満たされた時にエラーを投げる条件を定義することができ、マッピング処理が停止した時にユーザー定義のエラーメッセージをメッセージウィンドウにて表示することができるようになります

- ..\MapForceExamples\TimeService フォルダー内にある**Timeservice.mfp** プロジェクトにて、**getTimeZoneTime.mfd** ファイルエントリーをダブルクリックして、fault コンポーネントを含むサンプルを確認します。



Tマッピングでは、Web サービスフォルトがどのように使用されるか確認することができます（スクリーンショットでは WSDL 1.1 が示されます）。



上のサンプルではマッピング内で例外を定義する方法を確認できます。n0:timezone が "UTC" と一致しない場合に例外が発生します。

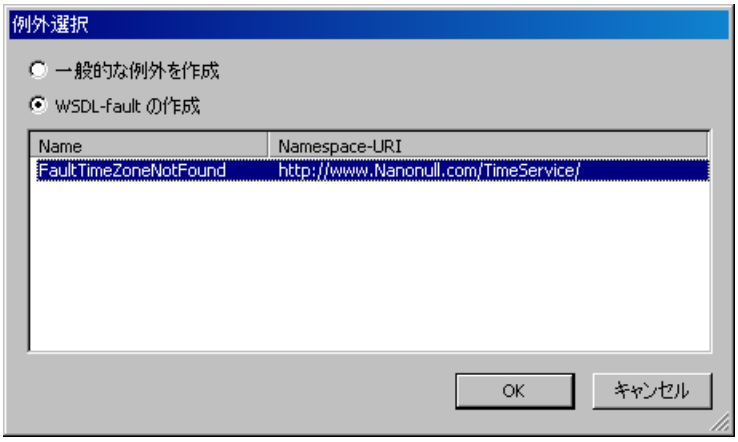
- **equal** コンポーネントにより、タイムゾーンが UTC と等しいかどうかがチェックされ、その結果 (bool) がフィルターコンポーネントへ渡されます。
- 条件が偽になった場合（UTC 以外のタイムゾーンが使用されている場合）、フィルターコンポーネントの on-false パラメーターにより FaultTimeZoneNotFound 例外がアクティブになり、マッピング処理が停止されます（それとは逆に on-true パラメーターを例外コンポーネントへ接続することもできます）。
- SoapFault メッセージには 2 つのエラーテキストが与えられます。

メモ:
この例ではフィルターコンポーネントが非常に重要な役割を果たします:

- **フィルター**コンポーネントにある**両方のパラメーター**(on-true とon-false)をマッピングしてください! そのうち 1 つをfault コンポーネントへマッピングして、もう 1 つをターゲットコンポーネントへマッピングすることで、フィルタリングされたソースデータの受け取りが行われます。このマッピングが行われない場合、fault コンポーネントがアクティブになることはありません
- **例外**ならびに**ターゲット**コンポーネントは**フィルター**コンポーネントへ**直接接続**されている必要があります。関数やその他のコンポーネントをフィルターと例外またはターゲットコンポーネント間に挿入することは許されません。

Web サービスへ fault を挿入する:

1. メニューオプションから「挿入 | 例外」を選択するか、ツールバーから例外アイコン をクリックします。
2. WSDL-fault の作成ラジオボタンをクリックして、リストからfault 名を選択します。「OK」をクリックすることで、fault コンポーネントが挿入されます。



Chapter 10

Web サービスの呼び出し

10 Web サービスの呼び出し

MapForce は、マッピング内からの直接の Web サービスの呼び出しをサポートします。これは、(MapForce 内の用語では、Web サービス関数と呼ばれる) Web サービス呼び出しをマッピングに挿入し、入力と出力コンポーネントを接続し、必要に応じて結果が使用できることを意味します (例えば、MapForce 出力ウィンドウ内でプレビューし、他のコンポーネントにパスし、ファイルに書き込むなど)。これにより、MapForce を、グラフィカルユーザーインターフェイスから簡単に構成することのできる強力な Web サービスクライアントとします。

MapForce は WSDL-スタイルの と REST-スタイルの Web サービスの呼び出しの両方をサポートします。ですから、Web サービスをマッピングに追加する際に、次のアプローチの 1 つを使用することができます:

- WSDL-スタイル Web サービスのために、呼び出される Web サービスの WSDL (Web サービス定義言語) を MapForce に与えます。wsdl ファイルが複数のサービス、エンドポイント、オペレーションを実装する場合、MapForce から直接複数のサービス、エンドポイント、オペレーションを選択や更新することができます。
- 非 WSDL Web サービスに関しては、Web サービスの詳細を MapForce に手動で入力します。これは、URL、リクエストメソッド (例えば、GET、POST、PUT)、リクエストとレスポンス構造 (XML または JSON スキーマとして)、およびパラメーターなどを含みます。オプションで、Web サービスの WADL (Web Application Definition Language) ファイルが呼び出される場合、WADL ファイルから定義をインポートすることができます。また、Web サービスのサンプル URL がある場合、手動で定義する必要がなく、MapForce に、URL を解析し、自動的にクエリ、テンプレートまたはマトリックスパラメーターを抽出するように命令することができます。

Web サービスを呼び出すには、マッピングの言語 (実行エンジン) を BUILT-IN に設定する必要があります。(次を参照してください: 変換言語の選択). MapForce により作成される Web サービスの呼び出しは、MapForce 自身により実行されるか、または、コマンドライン インターフェイス または、API 呼び出しなど MapForce Server を使用して、異なるマシン、または、プラットフォーム上で実行される可能性があります (http://www.altova.com/mapforce/mapforce-server.html)。

WSDL-スタイルの Web サービスに関しては、変換言語を C# と Java に追加設定することができます。このようにして、WSDL-スタイル Web サービス呼び出しは MapForce により生成された C# または Java コードによっても実行されることができます。

メモ: C# と Java では、デフォルトの HTTPS 認証オプションがサポートされます (すなわち、MapForce はサーバーの資格情報を確認します)。追加の HTTPS の設定 (すなわち、クライアントの資格情報) はこれらの言語ではサポートされません。

BUILT-IN を変換言語として選択する場合、セキュリティ保護なし Web サービスだけではなく、基本的な HTTP 認証またはデジタル証明書を必要とする HTTPS (TLS) 認証を必要とする Web サービスを呼び出すことができます。MapForce は、Web サーバーのデジタル証明書が有効かつ信頼できるかをチェックするオプションを与えます。

Web サービスの一部では、プリエンプティブな HTTP 認証を行うことができ、このオプションは MapForce から直接構成することができます。更に、WSDL-スタイルサービスでは、WS-Security への基本的なサポートを受けることができます。これにより、MapForce 内で **ユーザー名 トークン** と **タイムスタンプ** SOAP セキュリティトークンのコンテンツを構成することが可能です。

Web サーバーが反応しない場合、MapForce 構成して、タイムアウトする呼び出し回数を設定することができます。無効な URL などの他のサーバーエラーが報告されると、MapForce が処理を行います。

WSDL-スタイル Web サービスの場合、MapForce 内で次のプロトコールを使用することができます :

SOAP 1.1, SOAP 1.2	*RPC/encoded* と *document/literal* スタイルはサポートされます。Web サーバーが WSDL フォールトを返すと、マッピング実行は中断されます。このような場合、任意でマッピングエリアに例外コンポーネントを挿入して、エラーを処理します。Web サーバーが非 WSDL エラーを返した場合、マッピングの実行は中断され、エラーメッセージが返され

	ます (または、MapForceでマッピングをプレビューしている場合、メッセージはスクリーンに表示されます)。
HTTP GET	*url-encoded* スタイルはサポートされます。
HTTP POST	*url-encoded* と*text/xml* スタイルはサポートされます。

ジェネリックな (REST-スタイル) Web サービスの場合、HTTP 動詞を使用したHTTP プロトコールを介しての呼び出し (例えば、GET、POST、PUT) はサポートされます。 MapForce は、Web サービスのリクエストまたはレスポンス本文内のXML またはJSON のコンテンツをサポートします。他のコンテンツの種類はサポートしないことにご注意してください。

10.1 Web サービス呼び出しの追加 (REST- style)

このトピックでは、マッピングからジェネリックな (WSDL-スタイルではない) Web サービスを呼び出す方法について説明します。これは、 "REST" (また、通常 "RESTful" または REST-スタイル Web サービス) と称されるアーキテクチャのスタイルに従う Web サービスの大きなカテゴリを含みます。

必要条件

ジェネリックな HTTP Web サービスでは、メッセージの本文は、通常カスタム リクエストまたはレスポンスです。MapForce は、リクエスト内の JSON または XML データをサポートします。ですから、JSON または XML スキーマとして、MapForce に構造を与えることができる限り MapForce パースペクティブからは、XML または JSON 構造を必要とする、または返す、HTTP Web サービスを仮想的に呼び出すことができます。

Web サービスの XML または JSON スキーマは、サービス プロバイダー、または、WADL などの公式な言語、または、人間に読解可能な仕様で作成することができます。リクエスト/レスポンスの XML または JSON スキーマが存在する場合、Web サービスの呼び出しの作成は簡単です。WADL ファイルとして定義されている Web サービスが存在する場合、WADL ファイル* からインポートすることはできず、必要な調整を手動で行います。最後に、スキーマ ファイルが存在せず、XML または JSON インスタンス ファイルのみが存在する場合、XMLSpy (http://www.altova.com/ja/xmlspy.html) を使用してスキーマを作成、または、生成することができます。必要であれば、インスタンス ファイルを XML から JSON に、変換、またはその逆に変換することも可能です。

* WADL は、JSON 構造ではなく XML 構造のみを を定義する標準の方法を提供することにご注意してください。

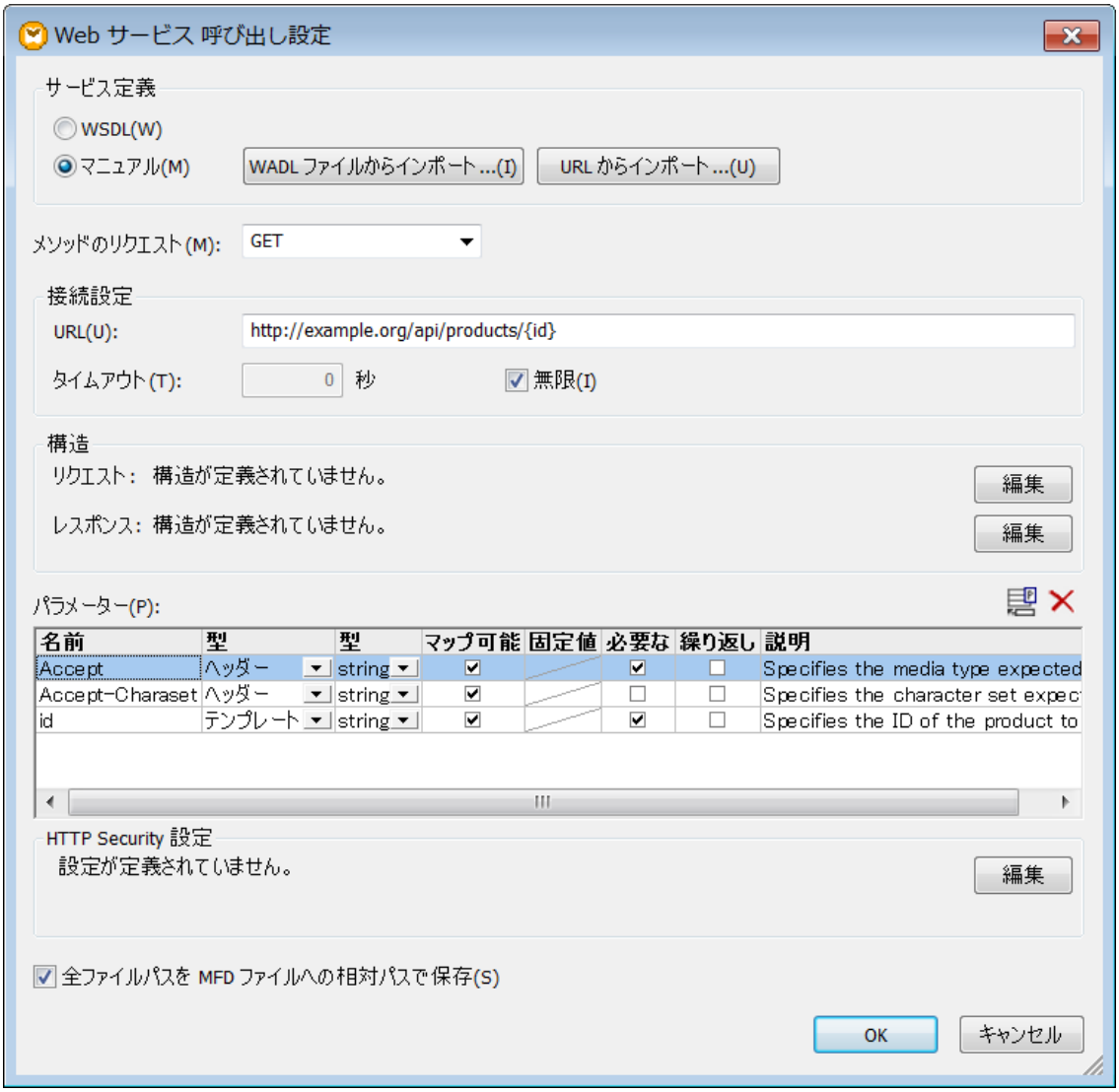
ジェネリックな Web サービスに呼び出しを追加する方法

1. **挿入** メニューから、**「Web サービス関数」** をクリックします。(または、**Web サービス関数の挿入** ツールバー ボタンをクリックします)。
2. サービスの定義から、**手動** を選択します。
3. 任意で、サービスを説明する WADL ファイルが存在する場合 **「WADL ファイルからインポート」** をクリックし、ファイルを選択します (WADL ファイルから Web サービス情報をインポートするを参照)。
4. MapForce が Web サービスを呼び出すために使用する HTTP リクエスト メソッドを選択します。既存のリストから値を選択するか、または、リクエスト メソッドの名前を入力します。 HTTP メソッドは大文字と小文字を区別します。
5. Web サービスの URL を入力します。Web サービスの URL がパラメーターを使用する場合、以下に注意してください:
   a. Web サービスを "テンプレート" または "matrix" スタイルのパラメーターで呼び出す場合、パラメーターを中かっこで囲みます。例えば :`http://example.org/api/products/{id}`。"パラメーター" テーブル内で各パラメーターの実際の設定を定義します。ランタイムで、 MapForce は、中かっこ内のパラメーター名を処理し、実際の値を含む最終 URL を生成します。
   b. "クエリ" URL パラメーターで Web サービスを呼び出す場合、 (例えば、`http://example.org/api/products?sort=asc&category=1&page=1`)、URL テキスト ボックス内にクエリの部分を入力せず、"パラメーター" テーブル内でのみパラメーターを定義します。
   サンプルに関しては、Web サービス パラメーターの定義 を参照
6. 任意で、**タイムアウト** の項目に、サーバーが応答しない場合、接続がタイムアウトする秒数を入力します。
7. HTTP メソッドが必要な場合、またはボディの部分 (XML または JSON 構造) を返す場合、 **構造** の下の **編集** ボタンをクリックし、ボディの部分の JSON または XML スキーマを参照します (Web サービスのレスポンスとリクエストの構造の定義を参照)。
8. Web サービスのパラメーターを定義します。任意で、 **URL からインポート** をクリックして、Web サービスのサンプル URL からパラメーターをインポートし、"パラメーター" テーブルを自動的に作成します (URL ファイルから Web サービス パラメーターをインポートするを参照)。パラメーターを URL からインポートした後、"パラメーター" テーブルのコンテンツを必要があれば変更することができます。更に詳しい情報に関しては、 Web サービス パラメーターの定義を参照してください。

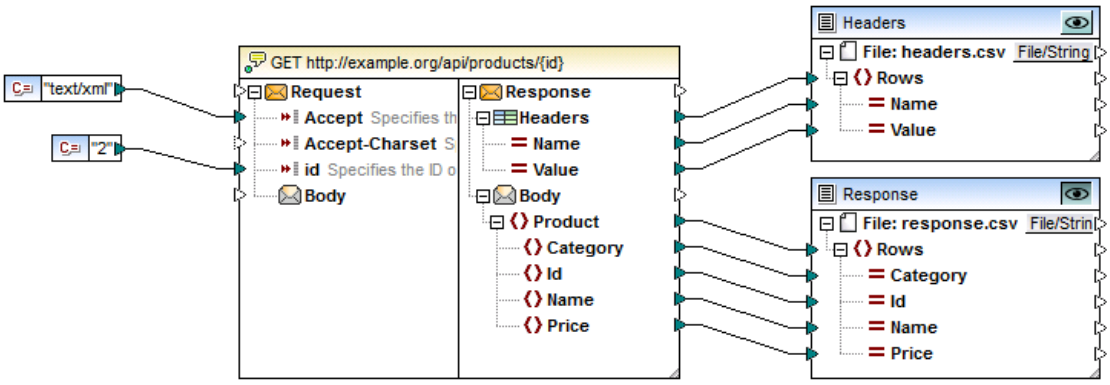
メモ: カスタムリクエストヘッダーを指定するには、スタイル "ヘッダー" を持つパラメーターを、パラメーター名がヘッダー名に対応し、また、パラメーター値がヘッダーの値に対応する箇所に追加します。更に、マッピング自身からヘッダーのリクエストの値を提供する場合、パラメーター型を "マップ可能" に設定します。

9. Web サービスが HTTP 認証 または 証明書 ベースのセキュリティを使用する場合、**HTTP セキュリティ設定** の下の**編集**ボタンをクリックし、必要とされるフィールドに入力します (HTTP セキュリティの設定を参照)。

「OK」をクリックした後、新規の Web サービス コンポーネントがマッピング エリアに追加されます。



下のマッピングは、GET リクエストを使用して、ID により製品を取得するために Web サービスを呼び出します。この特別な例では、HTTP リクエスト内で与えられた ID は定数の値 "2" を持ちます。しかしながら、マッピングのパラメーターにもなることができます。(次を参照 : マッピングにパラメーターを与える) または、MapForce によりサポートされる全てのコンポーネントにより与えられます。`id` パラメーターに加え、リクエストは、`Accept: text/xml` を含みます。リクエストとその値内で使用するために、ヘッダーを柔軟的に変更することができます (次を参照してください : サービス パラメーターの定義)。



この特定のマッピングは、2つの出力を持つように構成されています :

1) コンマで区切られた値としてのヘッダー
2) コンマで区切られた値としてのレスポンス本文

プレビューのために特定の出力を選択するには、コンポーネントの右上にある ボタンを押します。

Web サービス コンポーネントは2つのパートから構成されています :**リクエスト** と **レスポンス**。**リクエスト** のパートによりデータをマッピングからWeb サービスに与えることができ、また、**レスポンス** パートによりWeb サービスにより返されたデータにアクセスし、他のフォーマットにマップすることができます。リクエストとレスポンスの構成は、パラメーター、と呼び出し設定ダイアログボックスにより定義されたXML またはJSON リクエスト (またはレスポンス)構造により異なります (次を参照してください : サービスのレスポンスとリクエストの定義)。

(適用することができる場合)特定のリクエスト パラメーターを持つサービスを呼び出す場合、MapForce によりサポートされるコンポーネント間にマッピング接続を描きます (例えば、XML またはJSON ファイル)と **リクエスト** パート。同様に、サービスより返されたデータを他のフォーマットでマップする場合、レスポンスのパートと、MapForce によりサポートされる他のコンポーネント型の間にマッピング接続を描きます。MapForce の初心者であり、マッピングの接続に関する情報が必要な場合は、接続と作業するを参照してください。

("Content" で開始しない)追加ヘッダーである場合、サービスにより返されたレスポンスヘッダーもマップすることができます。ヘッダーの値は、"Name" と "Value" の2つの子アイテムを含む "ヘッダー" ( ) と呼ばれるアイテムを介して、サービス コンポーネントで使用することができます。この構造は、シーケンスとして振る舞い、サービスにより返される無制限の数のヘッダーからデータをマップすることを可能にします。レスポンスヘッダーからデータをMapForce によりサポートされている他のフォーマットにマップする場合、"ヘッダー" 構造ノードをマッピング内のターゲットシーケンスに接続します。例えば、**ヘッダー** シーケンス (とその子)をCSV コンポーネントの **行** シーケンス (とその子)に接続すると、CSV ファイル内の1行に対応します。

ジェネリックなHTTP Web サービス呼び出しを含むマッピングは他のマッピングと同様実行されることができます :

- MapForce を使用して、**出力** ボタンをクリックすることにより手動で行います。この場合、マッピングの呼び出しは、すぐに出力ペインに表示されます。
- MapForce Server (http://www.altova.com/mapforce/mapforce-server.html) を使用して、コマンドラインまたはAPI 呼び出しを使用します。これは、マッピングをマッピング実行可能ファイルに最初にコンパイルする必要があります (次を参照してください :MapForce Server ファイルにマッピングをコンパイルする)。
- 再帰的なジョブとして、MapForce Server がFlowForce Server の管理下で実行されています (http://www.altova.com/flowforce.html)。以下も参照してください :FlowForce Server にマッピングをデプロイする。

マッピングの実行は、以下のケースのエラーメッセージを伴い失敗する場合があります：

- Web サービスの呼び出しは 299 より大きい HTTP ステータスコードを返します。
- レスポンス XML または JSON を解析することはできません。
- Web サービスは、接続エラーまたは DNS resolution の問題により呼び出すことができません。

ジェネリックな Web サービスの呼び出しの手順を追っての説明の例は、以下を参照してください：

- 例：REST-スタイル Web サービスの呼び出し
- 例：RSS フィードからのデータのマッピング

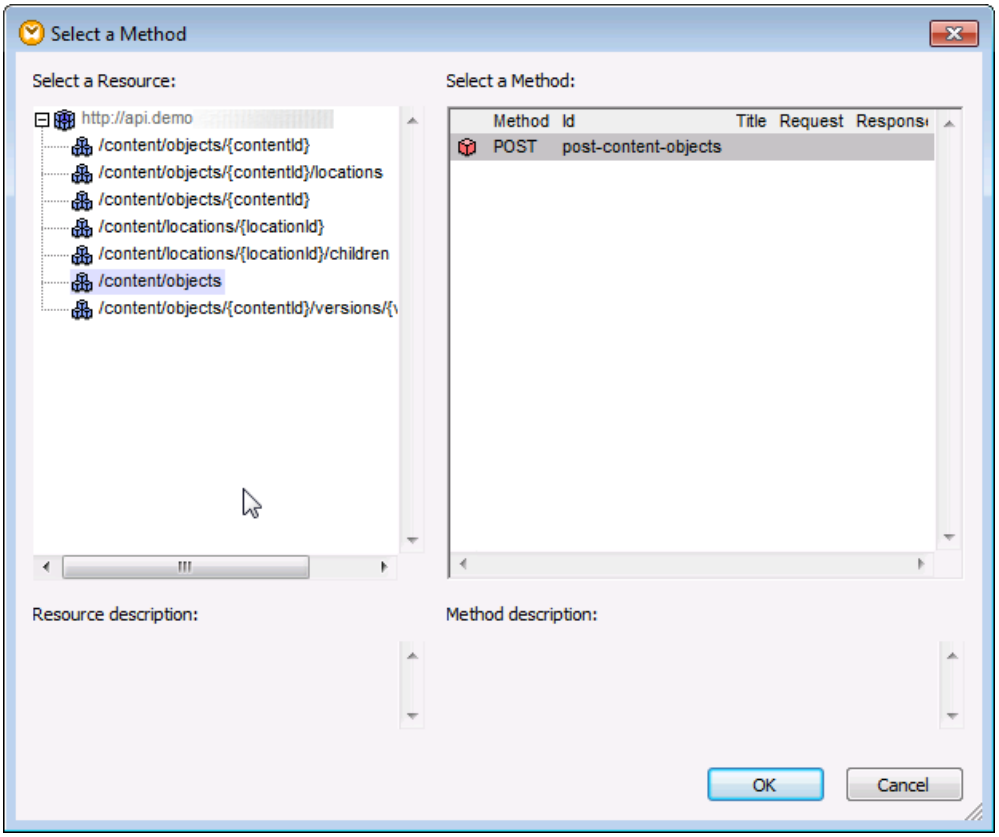
WADL ファイルから Web サービス情報をインポートする

Web Application Description Language (WADL) は、RESTful Web サービスを含む、Web アプリケーション内で使用されるレスポンスとリクエストのコンテンツを表す 1 つの方法です。MapForce は、WADL バージョン 2009 からの Web サービス情報のインポートをサポートします (http://www.w3.org/Submission/2009/SUBM-wadl-20090831/)。

サービスに WADL ファイルが存在する場合、手動で入力する代わりに、WADL ファイルからサービスの定義をインポートすることができます。WADL ファイルから情報をインポートする場合、サービス呼び出し設定ダイアログボックスで、パラメーターと XML メッセージ本文は自動的に作成されます。インポートの後、必要に応じて、パラメーター、および、レスポンス/リクエスト構造を手動で変更することができます。

WADL ファイルから Web サービス情報をインポートする：

1. Web サービス呼び出し設定ダイアログボックスを開きます (Web サービス呼び出し設定を参照)。
2. サービスの定義から、**「手動」**を選択します。
3. **WADL ファイルからインポート**をクリックし、WADL ファイルを参照します。



1. ダイアログボックスで、 リソースとメソッドを選択して、 **「OK」**をクリックします。

URL ファイルから Web サービスパラメーターをインポートする

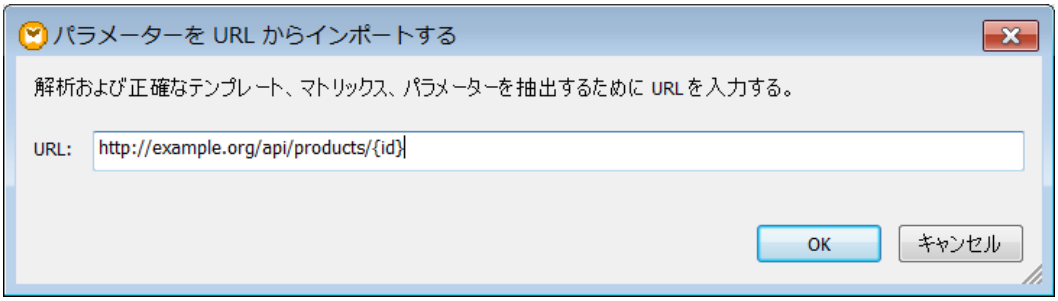
Web サービスの URL がパラメーターを使用する場合、MapForce に自動的に URL を解析して、パラメーターをインポートするように命令することができます。このオプションを使用すると、URL から抽出されたテンプレート、マトリックス、またはクエリパラメーターを、必要に応じて更に操作することのできる Web サービス呼び出し設定 ダイアログボックスの**パラメーター**テーブルで使用することができます。.

テンプレートパラメーターを使用する URL の例: `http://example.org/api/products/{id}`
マトリックスパラメーターを使用する URL の例 :`http://example.org/api/products;sort=asc;category=1;page=1`
パラメーターを使用する URL の例 :`http://example.org/api/products?sort=asc&category=1&page=1`

メモ: 解析のために、URL は "http://" または "https://" から開始する必要があります。

URL から Web サービスパラメーターをインポートする:

1. Web サービス呼び出し設定 ダイアログボックスを開きます (Web サービス呼び出し設定を参照)。
2. サービスの定義から、 **「手動」**を選択します。
3. **「URL からインポートする」**をクリックします。



1. テキストボックスにWeb サービスURL を入力または貼り付け、「**OK**」をクリックします。

Web サービスパラメーターの定義

URL パラメーターを使用するWeb サービスを呼び出す必要がある場合、パラメーターは、Web サービス呼び出し設定ダイアログボックスの "パラメーター" テーブル内で明示的に定義されている必要があります。"パラメーター" テーブルは、以下に表示されるように、Web サービスリクエスト内のカスタムヘッダーを定義するために使用されています。

手動または、任意で既存のWADL ファイルまたはURL からパラメーターを手動で入力することができます (WADL ファイルからWeb サービス情報をインポートするとURL ファイルからWeb サービスパラメーターをインポートするを参照してください)。必要な場合、インポートされたパラメーターが、更に変更することのできる**パラメーター**テーブル内で使用することができます。

Web サービスパラメーターを手動で追加または削除する:

1. Web サービス呼び出し設定ダイアログボックスを開きます (Web サービス呼び出し設定を参照)。
2. サービスの定義から、**「手動」**を選択します。
3. **パラメーターの追加** ( ) と**パラメーターの削除** (✕) ボタンをそれぞれクリックします。

パラメーターテーブル内の列には以下の意味があります:

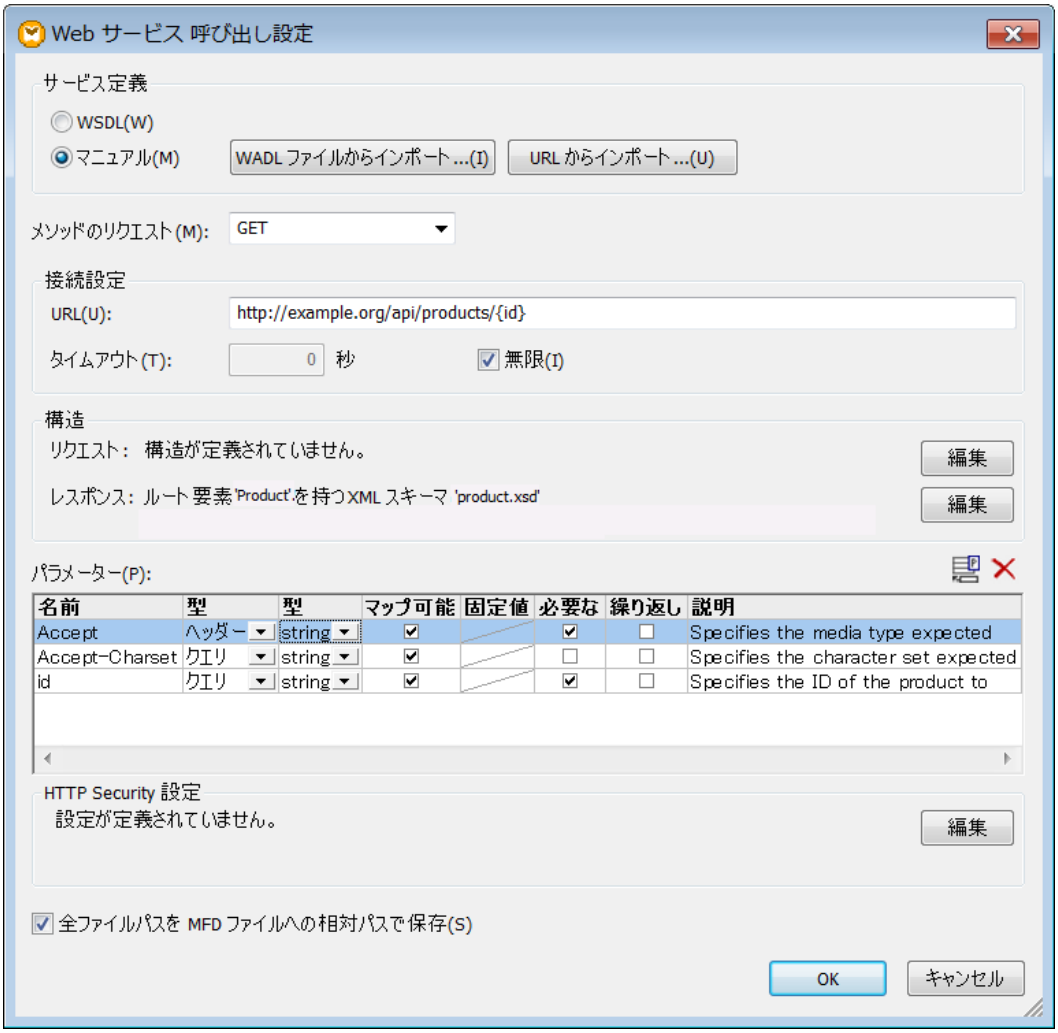
名前	URL パラメーターの名前を指定します。パラメーター名は一意で、英数字、数字、ピリオド (.)、ハイフン (-)、およびアンダースコア (_) により構成されている可能性があります。スペースはパラメーター名内では許可されていません。
スタイル	URL パラメーターの構文 (スタイル) を指定します。 サービスを呼び出す際、"ヘッダー" スタイルを使用してHTTP ヘッダーにパラメーターを追加します。例えば、`text/xml` の値を持つパラメーター`Accept` を追加することは、Web サーバーにMapForce がレスポンスがXML フォーマットであることを期待することを通知する`Accept: text/xml` ヘッダーを指定することと同じです。HTTP ヘッダーに関する詳細は、次を参照してください:http://www.iana.org/assignments/message-headers/message-headers.xhtml. キーと値のペアを以下のフォーマットを使用して定義するURL パラメーターのために "クエリ" スタイルを使用します:`?key=value&key=value` 例:`http://example.org/api/products?sort=asc&category=1&page=1` 中かっこで囲まれたURL パラメーターのために "テンプレート" スタイルを使用します。例:`http://example.org/api/products/{id}`。このようなパラメーター

	のために、MapForce は、RFC 6570 ルール (http://tools.ietf.org/html/rfc6570) に従い値をエスケープします。 キーと値のペアをフォーマットで定義するURL パラメーターのために、"マトリックス"スタイルを使用します :`;key=value;key=value;` 例 :`http://example.org/api/products;sort=asc;category=1;page=1;` ブール型 マトリックスパラメーターを使用するには、スタイルを "マトリックス"に、また、型を "ブール型 "に設定してください (次のオプションも参照してください)。
型	パラメーターのデータ型 (文字列、整数、日付など)を指定します。すべてのXML スキーマ型を使用することができます。サービスが実行される場合、文字列では内全ての値が文字列に変換されます。しかしながら、Web サービスを呼び出す際に間違った値を与えた場合、MapForce に変換エラーメッセージを表示させる場合、型の設定は意味があります。
マップ可能	マッピングからこのパラメーターへ値をパスする場合は、このチェックボックスを選択します。 このオプションは、"固定値 "オプションと相互排他関係です。
固定値	パラメーターの値を指定します。パラメーターが定数の値を持つ場合のみ適用できます。パラメータをマップすることができる場合は適用できません (前のオプションを参照)。
必須	パラメータがWeb サービスにより必要とされる場合、このチェックボックスを選択します。必要であり、マップすることのできるパラメーターに対しは、MapForce は検証チェックを行います (すなわち、パラメーターに値がない場合、エラーメッセージは表示されません)。
繰り返し	パラメーターが単一値、または、複数の値かを指定します。この設定は、マップすることのできるパラメーターに対してのみ適用することができます。単一パラメーターを使用して、同じサービスの呼び出しの中で複数の値をパスすることが可能になります。 チェックボックスを選択すると、値のシーケンスを、単一の値の代わりに、パラメーターの構造ノードにマッピング上で接続することができます。MapForce は、パラメーターのスタイルに従い、値のシーケンスを処理します : • "テンプレート"パラメーターに対しては、値はカンマで区切られたサービスとして与えられます。例 :`http://example.org/api/products/1,2,3` • "クエリ"パラメーターに対しては、パラメーター名は各値のために繰り返されます。例 :`http://example.org/api/products?color=red&color=green&color=blue` • スタイルが"マトリックス"の場合、複数の値は、カンマで区切られます。例 : `http://example.org/api/products;color=red;color=green;color=blue;size=big;size=small;` • スタイルが "ヘッダー"の場合、HTTP ヘッダーは各値のために繰り返されます。
説明	任意でパラメーターの詳細を指定します。パラメーターをマップすることができる場合、この項目に入力される詳細は、マッピングコンポーネントにマッピングアイテムの横の注釈として表示されます。

例 1

下で説明されているサービスは、HTTP GET メソッドを使用して、製品を識別子 (id) に従い取得します。Web サービスの URL は、中かっこ内の id パラメーターを指定します。id パラメーターは、パラメーターテーブル内にも存在し、スタイル "テンプレート" を持ちます。また、**マップ可能** でもあり、これによりパラメーターが (例えば、データベース、ファイル、または定数から取得することのできる) id の実際の値を接続することのできる、マッピング内の構成ノードとして表示されます。実行ランタイムのマッピングでは、このパラメーターは実際の値と置き換えられます。ですから、値が "1" の場合、URL は `http://example.org/api/products/1` になります。

定数 id 値を与える場合、オプション "マップ可能" のチェックを解除し、マッピングから値を与える代わりに、値を "固定値" の列に入力することができます。



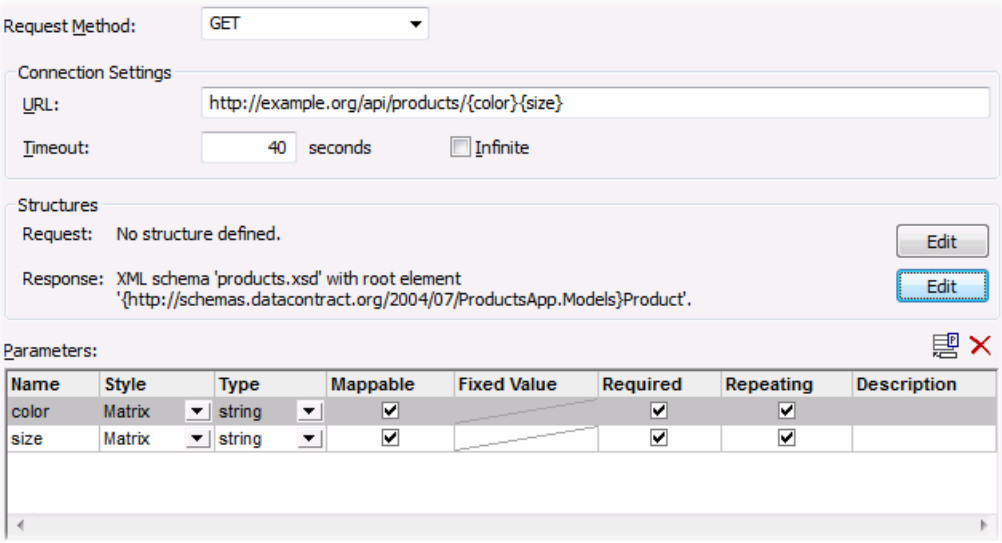
パラメーター Accept と Accept-Charset は "ヘッダー" スタイルを持ちます。これらのパラメーターは、カスタムリクエストヘッダーを持つサービスを呼び出すために使用されます。ヘッダー値を与えるには以下の2つの方法があります：

- **「マップ可能」** をチェックし、マッピングからカスタムヘッダー値を与えます。
- オプション **「マップ可能」** のチェックを解除し、"固定値" 列に値を直接入力します。

例 2

下で説明されているサービスは、引数として与えられた色とサイズに一致する製品のリストを取得します。パラメーターのスタ

イルは、"マトリックス"ですので、URL とマッピングテーブル内のプレースホルダーとして定義されます。パラメーターをマップすることは可能であり、"繰り返し"オプションがチェックされています。これは、値がマッピング上の値のシーケンスより読み込まれ (例えば、テキストファイル内の行のリスト、XML ノード、またはデータベース列) ランタイムにサービスに与えられます。このため、以下のような URL は、色として、`red` と `blue` が与えられ、サイズとして `big` と `small` が与えられており、`http://example.org/api/products/;color=red;color=blue;size=big;size=small` となります。



URL を `http://example.org/api/products;color=red,blue;size=big,small` にする場合、以下を行います:

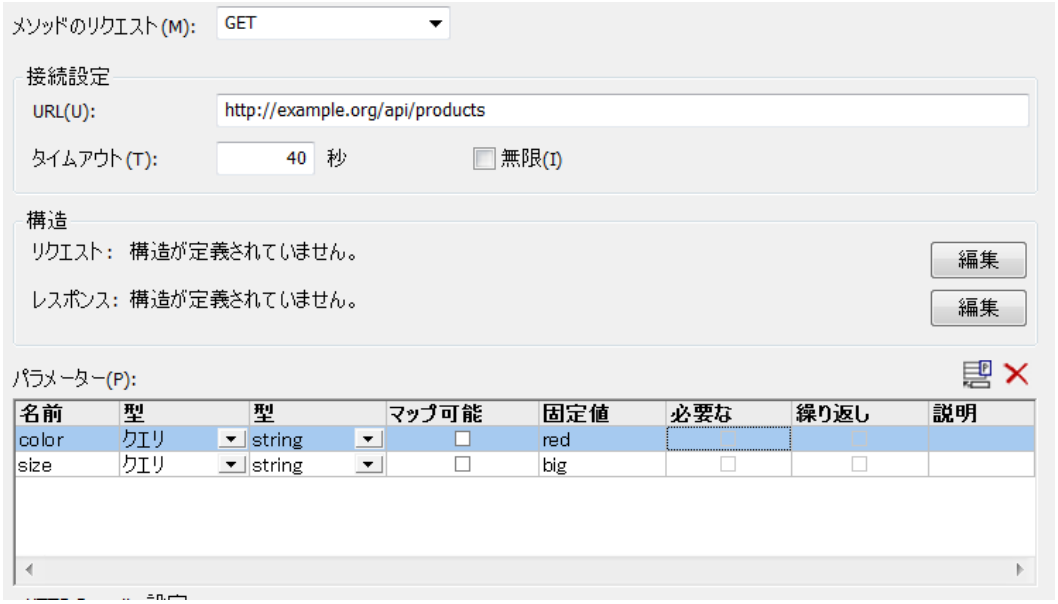
1. URL を `http://example.org/api/products;color={color};size={size}` として入力します
2. "パラメーター"テーブルのパラメーターのスタイルを "テンプレート"に変更します。

URL を `http://example.org/api/products?color=red,blue&size=big,small` にする場合、以下を行います:

1. URL を `http://example.org/api/products?color={color}&size={size}` として入力します
2. "パラメーター"テーブルのパラメーターのスタイルを "テンプレート"に変更します。

サンプル 3

下で説明されるWeb サービスは、引数により与えられた色とサイズに一致する製品のリストの抽出に関して説明しています。今回は、スタイル"クエリ"のみを使用しています。このスタイルのためには、URL 内のプレースホルダーとしてパラメーターを定義する必要はありませんので、内のみで定義されています。 "固定値"の下の"パラメーター"テーブルに直接入力されるので、 "マップ可能"オプションはチェックされていません。マッピングランタイムでは、下のURL は`http://example.org/api/products?color=red&size=big`。



Web サービスのレスポンスとリクエスト構造を定義する

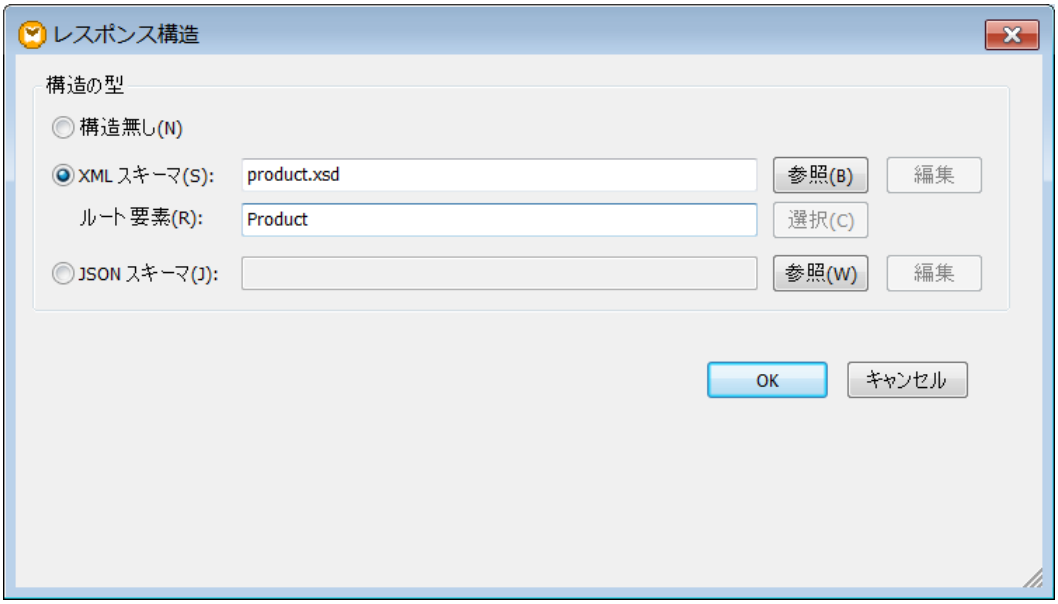
POST またはPUT などのHTTP リクエストを使用するWeb サービスのためには、XML またはJSON コンテンツをリクエストメッセージのボディの部分に含むことができます。同様に、Web サーバーは、API 呼び出しのレスポンス内のXML またはJSON コンテンツを返す可能性があります。このような場合、you must instruct MapForce にHTTP メッセージのボディのスキーマが何かを命令する必要があります。このようにして、構造により定義された要素が、マッピングにWeb サービスを追加した後にマップするためのアイテムとして表示されます。

リクエストまたは レスポンス構造のXML または JSON スキーマを定義する：

1. クエスト または レスポンス構造のXML または JSON スキーマをWeb サービスから取得します。

ヒント： スキーマファイルを持たない XML または JSON インスタンスファイルの場合 XMLSpy を使用してスキーマファイルを生成することができます。必要であれば、XMLSpy は、インスタンスファイルをXML からJSON へ、または、インスタンスファイルをJSON からXML へ変換することができます。

2. Web サービス呼び出し設定 ダイアログボックスを開きます (Web サービス呼び出し設定 を参照)。
3. サービスの定義から、 **手動** を選択します。
4. **構造** の下の **リクエスト** の横の **編集** (または場合によっては、 **レスポンス**) をクリックします。
5. ダイアログボックスで、 **参照** をクリックして、XML またはJSON スキーマファイルへのパスを入力します。



XML スキーマの一部は (すなわち、親がスキーマ要素である) グローバル宣言付きの要素を定義します。スキーマのようなスキーマのために、MapForce 内のマッピング構造のルート要素になる要素を選択することができます。これを行うには、**参照** をクリックし、表示されるダイアログボックス内で、希望するルート要素を選択します。

10.2 Web サービス呼び出しの追加 (WSDL- Style)

このトピックでは、WSDL-スタイルWeb サービスへ呼び出しをマッピングする方法について説明します。ジェネリックな Web サービスをマッピングに追加する方法に関しては、Web サービス呼び出しの追加 (ジェネリックな) を参照してください。

マッピングエリアに WSDL-スタイルのWeb サービスの呼び出しを追加する前に、接続するWeb サービスを説明する Web サービス定義言語 (wsdl) が搭載されていることを確認してください。

マッピングの呼び出しのためにWeb サービスを追加する:

1. **挿入**」メニューから「**Web サービス関数**」をクリックします。(または、「**Web サービス関数の挿入**」 ツールバーボタンをクリックします)。
2. サービスの定義から、このオプションが既に選択されていない場合、**WSDL** をクリックします。.
3. WSDL 設定から、**参照**」をクリックし、Web サービスの定義言語 (wsdl) ファイルを選択します。WSDL (1.0 と 2.0) のバージョンそれぞれ1つずつの2つのサンプル wsdl ファイルが、MapForce Example フォルダー内にあります。対応するパスは、以下のとおりです: **<Documents>\Altova\MapForce2017\MapForceExamples\TimeService\TimeService.wsdl** と**<Documents>\Altova\MapForce2017\MapForceExamples\TimeServiceWsdl2\TimeService20.wsdl**.
4. .wsdl ファイルが複数のサービス、オペレーションとエンドポイントを定義する場合、**選択**」をクリックして、必要なオプションを選択します。
5. 必要であれば、他の設定を定義します (WSDL 呼び出し設定を参照してください)。
6. 「**OK**」をクリックします。

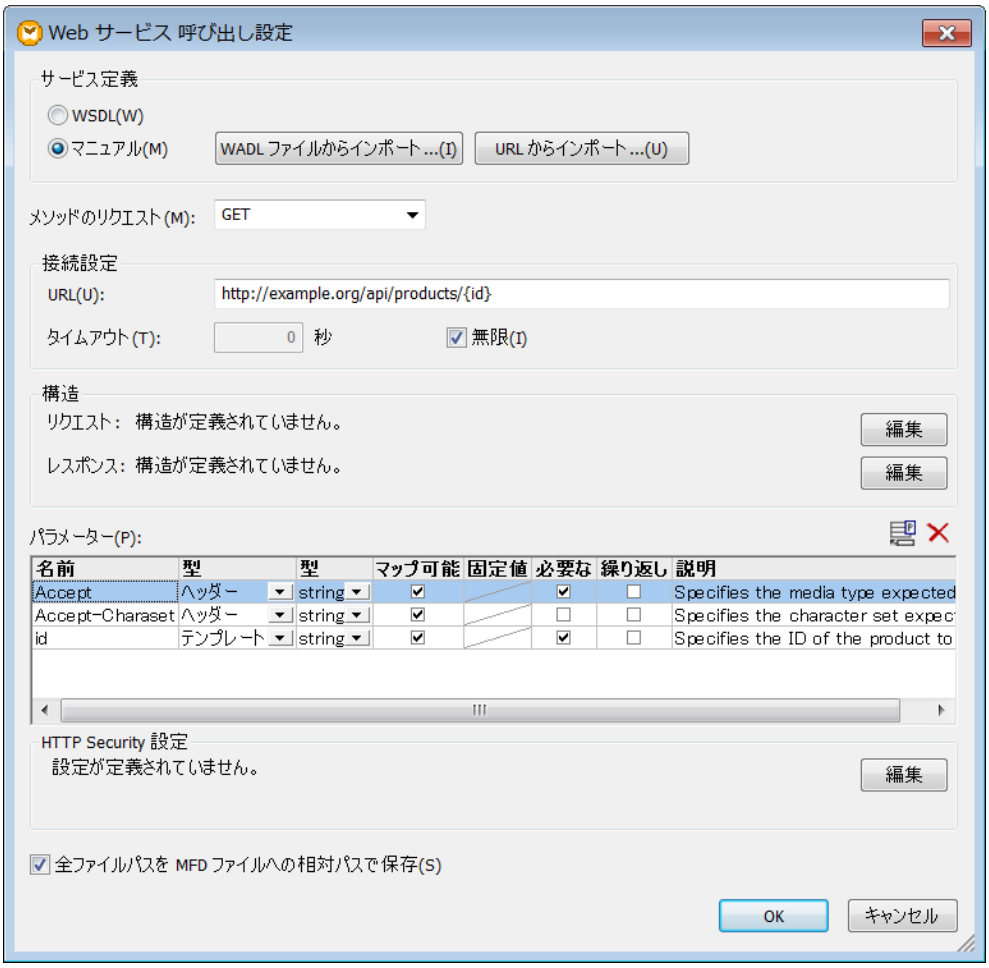
サンプル:WSDL-スタイルWeb サービスの呼び出しも参照してください

10.3 WSDL 呼び出し設定

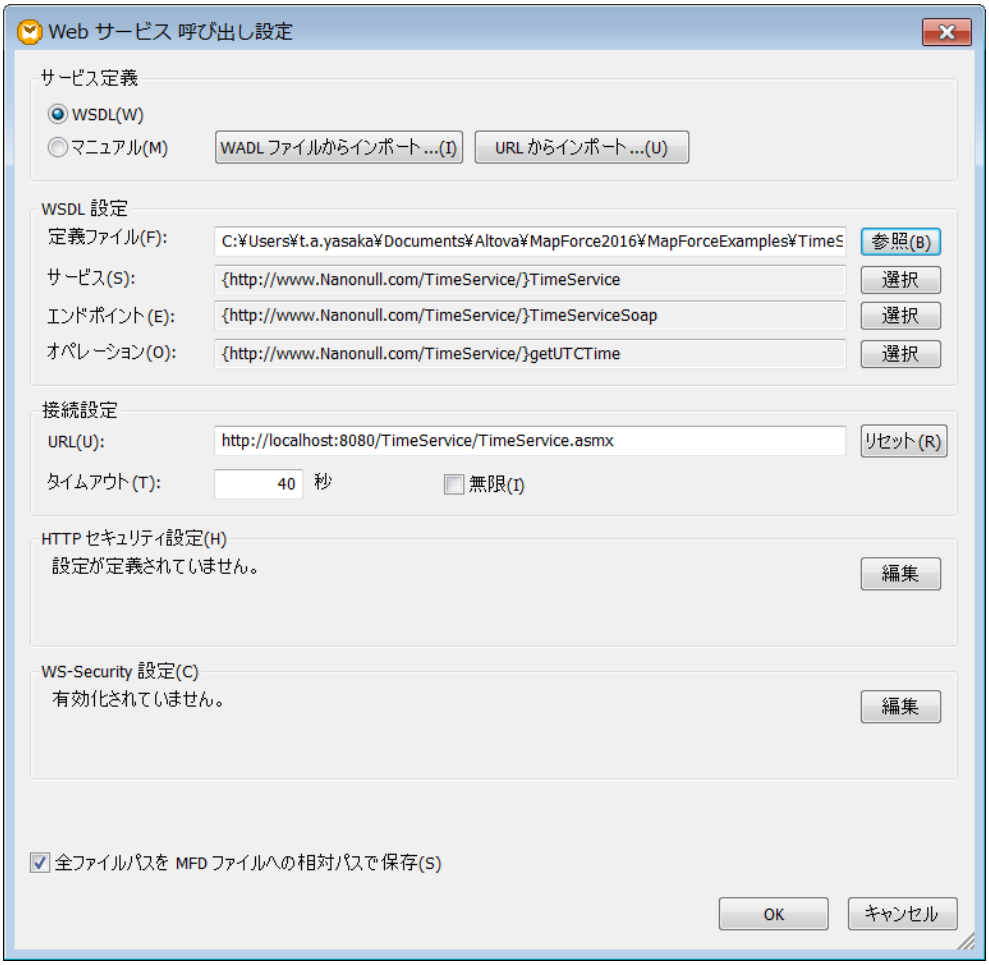
Web サービス呼び出し設定 ダイアログボックスからWeb サービスに適用することのできる設定を変更することができます。このダイアログボックスを開くには、以下を行います：

- マッピングのWeb サービス コンポーネントをクリックし、 **コンポーネント** メニューから **「プロパティ」** をクリックします。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、 **「プロパティ」** をクリックします。

ダイアログボックスの上の **サービス定義** オプションが **WSDL** または **手動** に設定されているかにより使用することのできる設定のリストは異なります。次のイメージはジェネリックなHTTP Web サービスに適用することのできる設定を表しています：



次のイメージはWSDL-スタイルのWeb サービスに適用することのできる設定を表しています：



使用することのできる設定は下で説明されています。

サービスの定義

定義する設定がWSDL-スタイルWeb サービスに適用される場合、 **WSDL** を選択し、それ以外の場合、**手動**を選択します。

手動が選択されている場合、Web サービスの設定をダイアログボックスに手動で入力する必要があります。オプションで、Web サービスのWADL ファイルが存在する場合、Web サービスの設定を**WADL からインポート**をクリックすることでインポートできます (WADL からWeb サービス情報をインポートする) を参照。また、URL からWeb サービスパラメーターを抽出する場合は、**URL からインポート**ボタンをクリックします (URL ファイルからWeb サービスパラメーターをインポートするを参照)。

WSDL 設定

WSDL-スタイルWeb サービスのみに適用することのできる設定のグループ

WSDL 定義	呼び出されるWeb サービスの Web サービス定義言語 (WSDL) ファイルを指定します。

サービス	呼び出されるWeb サービスの名前を指定します。WSDL ファイルが複数のWeb サービス複数のWeb サービスを定義する場合、 **選択** をクリックして、必要なWeb サービスを選択します。
エンドポイント	呼び出されるWeb サービスのエンドポイント (またはポート) を指定します。選択されたWeb サービスの名前が複数のWeb サービスエンドポイントを定義する場合、 **選択** をクリックして、必要な項目を選択してください。
オペレーション	呼び出されるWeb サービスのオペレーションを指定します。選択されたエンドポイントが複数のWeb サービスオペレーションを定義する場合、 **選択** をクリックして、必要な項目を選択してください。

リクエストメソッド

この設定はジェネリックな (WSDL ではない)Web サービスのみに適用することのできる設定です。これはサービス (例えば、GET, POST, PUT など) により使用されるHTTP メソッド (動詞) を定義します。

接続設定

URL はWeb サービスのアドレス (URL) を指定します。**タイムアウト** は、サーバーからのレスポンスが内場合、Web サービスの呼び出しがタイムアウトされる間隔を定義します。呼び出しがレスポンスを無制限の時間待つ場合は、**無限** を選択してください。

パラメーター

ジェネリックな (WSDL ではない)Web サービスのみに対してこのグループの設定を適用することができます。**パラメーター** テーブルは、MapForce がWeb サービスを呼び出すために使用するURL パラメーター を指定します (Web サービスパラメーターの定義を参照)。

構造

この設定グループはジェネリックな (WSDL ではない)Web サービスに適用することができます (Web サービスのレスポンスおよびリクエスト構造の定義を参照)。

HTTP セキュリティ設定

Web サービスが (HTTP または HTTPS) 認証 を必要とする場合、必要な認証設定を指定するために **編集** をクリックします (HTTP セキュリティの設定を参照してください)。

WS-Security 設定

Web サービスがカスタム SOAP セキュリティを使用する場合、 **編集** をクリックして設定の詳細を指定してください (WS-Security の設定を参照してください)。

MFD ファイルに対して相対的な全てのファイルパスを保存する

このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックス内に表示される

MapForce Design (.mfd) ファイルの場所に対する相対ファイルパスを保存します。 相対と絶対パスを使用も参照してください。

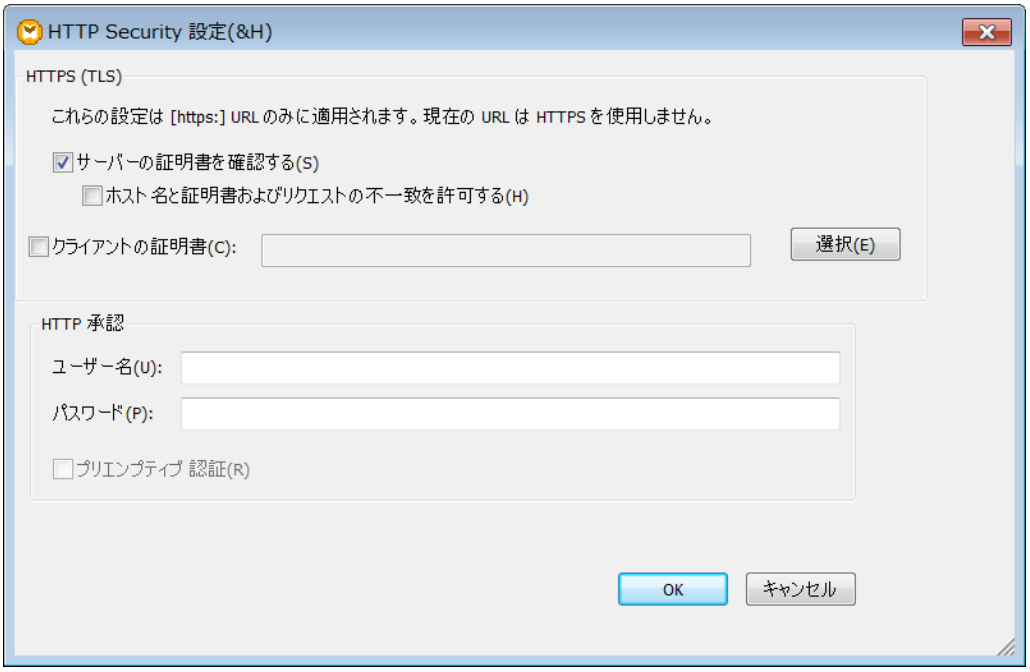
10.4 HTTP セキュリティの設定

以下の場合は、HTTP セキュリティ設定 を構成する必要があります :

- HTTPS を介して Web サービスを呼び出すには、クライアント証明書が必要です。
- サービスは有効でないサーバー資格情報を使用しています。
- Web サービスは HTTP 認証を必要とします。

HTTP セキュリティの設定 :

1. 「WSDL 呼び出し設定」ダイアログボックスを開きます。 (WSDL 呼び出し設定を参照してください)。
2. HTTP セキュリティ設定の横の **「編集」** ボタンをクリックします。



HTTP セキュリティ設定 ダイアログボックス

オプションの "HTTPS (TLS)" グループは HTTPS を介して呼び出される Web サービスに適用されます。

サーバー証明書のチェック	デフォルトでこのチェックボックスが選択されており、リクエストを処理する前に、MapForce は、サーバーの資格情報をチェックするように構成されています。このオプションが有効化されると、Web サービスリクエスト (とマッピング) サーバーが信頼できない場合、または使用中のオペレーティングシステムが Web サーバーを信頼するように構成されていない場合は失敗します。 特別な理由がない限り、このオプションをスイッチオフすることは奨励されません。 以下も参照 : クロスプラットフォーム証明書の管理。
証明書とリクエストのホスト名の	特定のホスト名のために発行されたサーバー証明書 (例 :

不一致を許可する	*www.example.com*) が異なったホスト名 (例 : *example.com*) インストールされている場合に使用することができます。 証明書のホスト名がWeb サービスのホスト名に一致しない場合でも、チェックボックスを選択して、認証 を続行してください。
クライアント証明書	**現在のユーザー\パーソナル** 証明書 ストアからクライアント証明書を**選択**します。store をクリックします。これは、クライアント証明書が**現在のユーザー\パーソナル** 証明書ストアに既に存在していると仮定しています。それ以外の場合、 証明書のスナップインを使用してインポートしてください (Windows 証明書ストアにアクセスするを参照してください)。 マッピングが他のオペレーティングシステムへの実行のためにデプロイされる場合は、同じ証明書がターゲットオペレーティングシステムにもインストールされている必要があります。更に詳しい情報に関しては、クロスプラットフォーム証明書の管理 を参照してください。

Web サービスがHTTP 認証を必要とする場合、オプションの "HTTP 認証 "グループが適用されます。

ユーザー名	Web サービスにアクセスするために必要なユーザー名を指定します。
パスワード	Web サービスにアクセスするために必要なパスワードを指定します。
プリエンプティブ認証	Web サービスが、承認データを最初の呼び出しで期待する場合、このチェックボックスを選択します。 それ以外の場合、MapForce は、ユーザー名とパスワード無しにアクセスを試み、サーバーが認証を必要とすると、ユーザー名とパスワードを使用します (HTTP ステータス 401)。

10.5 WS- Security の設定

WS-Security 設定は、Web サービスがWS-Security により保護されていて、**ユーザー名 トークン** セキュリティトークンの情報を必要とする場合構成される必要があります。

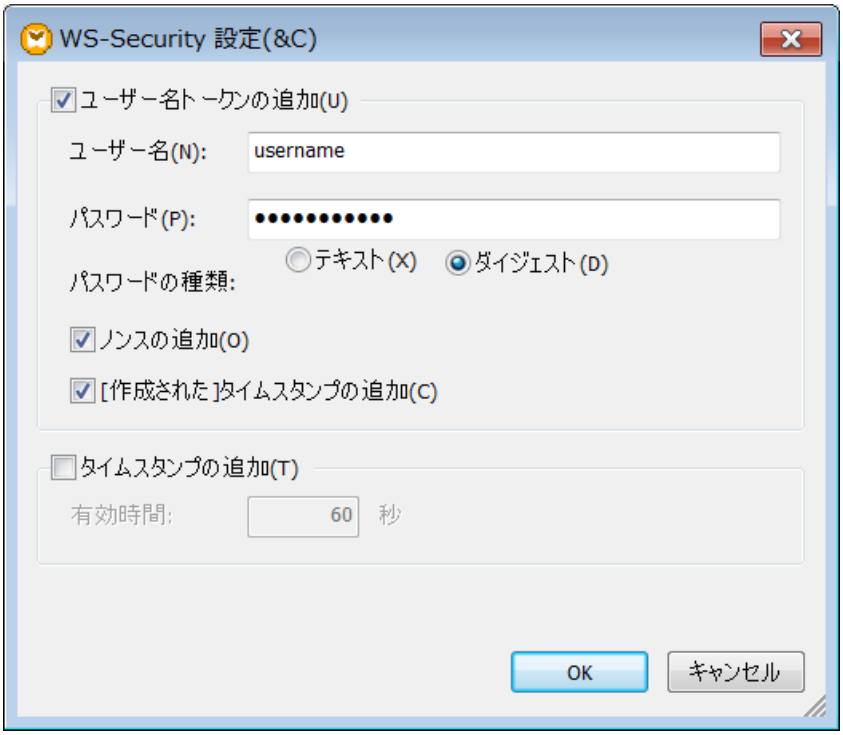
規約

Web サービスへ適用することのできる名前空間の省略名称がこのトピックで使用されています：

プレフィックス	名前空間
wsse	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd`
wsu	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd`

WS-Security の設定方法：

1. 「WSDL 呼び出し設定」ダイアログボックスを開きます。（Web サービスの呼び出しの設定を参照してください）。
2. WS-Security 設定の横の **「編集」** ボタンをクリックします。



WS-Security 設定ダイアログボックス

ユーザー名トークンの追加	ユーザー名 トークンは、SOAP メッセージのヘッダーに存在する

	WS-security の任意の要素です。ユーザー名 トークンは、Web サービスのコーラーを認証するために Web サービスにより使用されます。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken`
ユーザー名	ユーザー名 トークン内に含まれるユーザー名を入力します。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Username`
パスワード	ユーザー名 トークン内に含まれるパスワードのテキストを入力します。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Password`
パスワードの種類	ユーザー名 トークン内に含まれるパスワードの種類を選択します。Web サーバーがこのモード内でパスワードを期待する場合は、**ダイジェスト** を選択します。それ以外の場合は、**テキスト**を選択します。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Password/@Type`
ノンスの追加	ユーザー名トークンにノンスを追加する場合、このチェックボックスをチェックします。ノンスは、追加セキュリティを追加するために各ユーザー名トークンを一意に識別するランダムな値です。このオプションを有効化する場合、**"作成された"タイムスタンプを追加** オプションも有効化することが奨励されます。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Nonce`
"作成された"タイムスタンプの追加	各ナンスのためにタイムスタンプを追加するためにこのオプションを選択します。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsu:Created`
タイムスタンプの追加	SOAP メッセージのための有効時間 (TTL) 値を有効化するにはこのチェックボックスをチェックします (次のオプション参照)。 **wsse** 仕様内で (規約を参照) このフィールドは以下に対応します:

	/wsu:Timestamp
有効時間	*メッセージの*取得*または*返信*の*可能性を削減*するため*に、 SOAP *メッセージのための*有効時間 (TTL) を入力します。

10.6 例 :REST- スタイル Web サービスの呼び出し

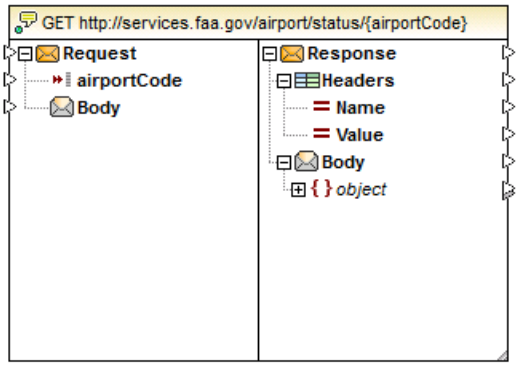
この例は、MapForce から、ジェネリックな HTTP Web サービスを呼び出す方法について説明しています。この例で呼び出される Web サービスは以下で見つけることができます :http://services.faa.gov/docs/services/airport/#airportStatus 。この Web サービスは、例えば、"SFO"、"IAD"、"ABE"、"DFW" などの 3つの英数文字からなる空港名コードを引数として受け入れ、米国の主要な空港の現状を XML または JSON 構造として返します。この例は、次のパスで見つけることのできるマッピングデザインファイルが伴われます :: **<Documents>\ Altova \ MapForce2017\ MapForceExamples\ AirportStatus.mfd**。

この例では、Web サービスのレスポンスは、JSON フォーマットで、JSON ファイルにマップされます。ですから、Web サービスのレスポンス構造の JSON スキーマが必要になります。JSON スキーマは Web サービスプロバイダーにより提供されたサンプルレスポンスより XMLSpy を使用して生成されています。このスキーマをマイドキュメントフォルダーに相対的な次のパスで見つけることができます :**<Documents>\ Altova\ MapForce2017\ MapForceExamples \ AirportStatus.schema.json**。

リクエストパートに関しては、リクエストに応じて "SFO" の値を提供する Web サービスを呼び出します。レスポンスパートに関しては、レスポンスデータを JSON ファイルにマップします。ですから、マッピングは、次の 3つの主要なコンポーネントから構成されます :Web サービス呼び出し、入力パラメーター、JSON 出力ファイル。

ステップ 1 : Web サービスコンポーネントを追加する

1. マッピングの変換言語が BUILT-IN に設定されていることを確認してください (変換言語の選択 を参照する)。
2. **「挿入」** メニューから、**「Web サービス関数」** をクリックします。
3. **「サービスの定義」** から **「手動」** をクリックします。
4. リクエストメソッドを GET に設定し、URL を `http://services.faa.gov/airport/status/{airportCode}` に設定します。中かっこ内の値は、ランタイム時に実際の空港名コードを置き換えられるテンプレートパラメーターです (Web サービス呼び出しの追加 (REST- スタイル) を参照)。
5. **「パラメーターの追加」** ( ) ボタンをクリックして、新規のパラメーターを "パラメーター" テーブルに追加します。パラメーターの名前は、URL 内の中かっこ内で指定されているパラメーターと同じである必要があることに注意してください。スタイルを "テンプレート" に設定し、"文字列 " を選択し、"マップ可能 " と "必須 " オプションをチェックします。"説明 " フィールドは任意です。
6. Web サーバーにクライアント (つまり、MapForce) がレスポンスに JSON を期待することを伝えるために、ヘッダーを追加します。これを行うには、**「パラメーターの追加」** ( ) ボタンを再度クリックし、パラメーターを "Accept" と名づけ、スタイルを "ヘッダー " とし、**application/json** を固定値として入力します。
7. **「レスポンス」** から **「編集」** ボタンをクリックして、Web サービスレスポンスのためのスキーマを参照します。Web サービスレスポンスのためのスキーマを参照します。スキーマは以下のパスで検索することができます : **<Documents>\ Altova\ MapForce2017\ MapForceExamples \ AirportStatus.schema.json**.
8. **「OK」** をクリックします。

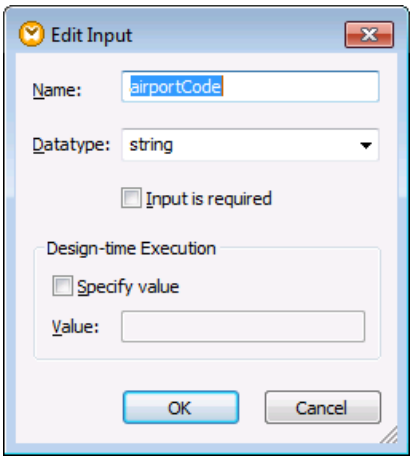


この段階では、Web サービスは、マッピングエリアで使用することができ、レスポンスパートからリクエストするためにデータをマッ

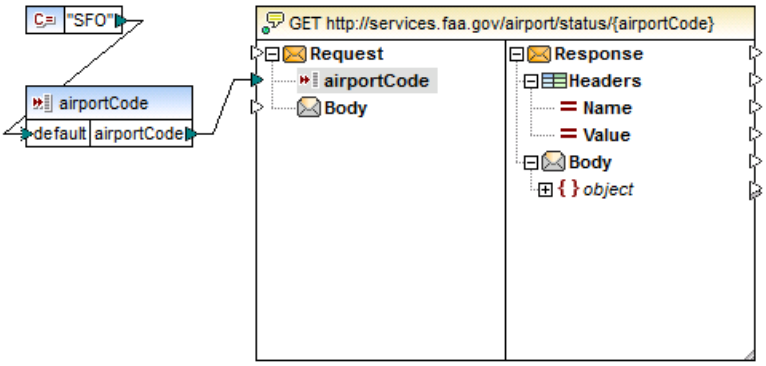
プすることができます。

ステップ 2 : 入力パラメーターの追加

1. **挿入** メニューから **入力の挿入** をクリックします。
2. "airportCode" をパラメーター名として入力し、 **入力が必要です** オプションをクリックして、チェックを解除します (上に示されるように、入力は定数から与えられます)。



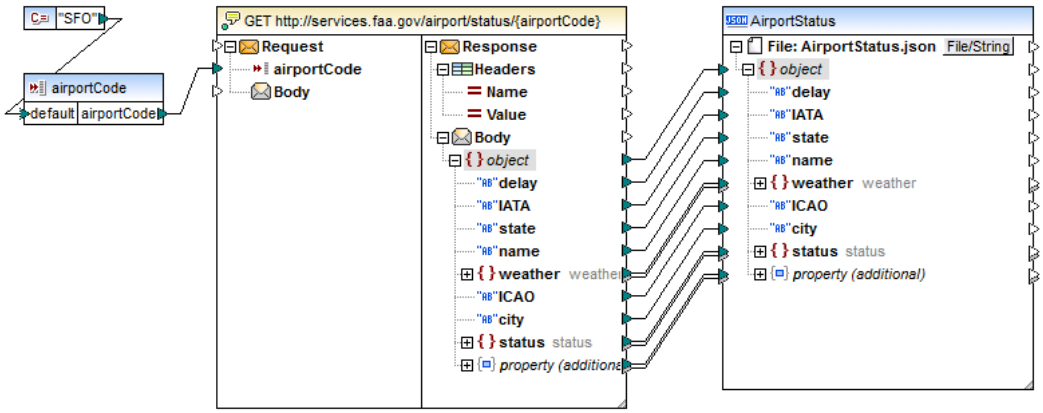
3. (メニューコマンド **挿入 | 定数**) を使用して "SFO" の値を持つ定数を追加し、パラメーター入力と接続します。
4. aiportCode パラメーターの出力を上に表示されるようにリクエストの入力に接続します。



この段階では、Web サービスのリクエストパートは、完成しています。マッピングランタイムにおいて、Web サービスは定数により与えられた値 (この場合、 "SFO") により呼び出されます。

ステップ 3 : JSON 出力の追加

1. **挿入** メニューから **JSON スキーマファイル** をクリックします。
2. **<Documents>\ Altova\ MapForce2017\ MapForceExamples \ AirportStatus.schema.json** を参照して、 **開く** をクリックします。
3. サンプル JSON ファイルを与えるようにプロンプトされると、 **スキップ** をクリックします (サンプル JSON ファイルは生成されるため、必要ありません)。
4. **接続** メニューで、 **一致する子を自動的に接続** メニューアイテムが選択されていることを確認してください。このオプションを有効化することにより、次のステップで時間を節約することができます。
5. Web サービスのレスポンス本文の `object` ノードを、JSON コンポーネントの `object` ノードに下に示されるように接続します。 **一致する子を自動的に接続** が前のステップで有効化されているため、全ての子孫ノードが自動的に接続されるため、各接続を描く必要がありません。



ステップ 4：マッピングの実行

Web サービスを呼び出す準備が整うと、**「出力」**ボタンをクリックして、マッピングを実行し、一般的な出力をプレビューします。Web サービス呼び出しに成功すると、**「出力」**タブは、返された JSON 構造を表示します。呼び出しに失敗すると、MapForce は、エラーを状況に応じて返します。Web サービスの説明にも記述されている通り、エラー 502（ゲートウェイが不適切です）がサーバーが存在しない場合、Web サービスにより返される可能性があります。Web サービスによりサポートされていないカスタム空港名コードを使用した場合、HTTP "見つかりません" (404) エラーが発生する可能性があります。

MapForce Server が存在する場合、（http://www.altova.com/ja/mapforce/mapforce-server.html）マッピングをマッピング実行可能ファイル（*.mfx）にコンパイルし、コマンドライン、または、MapForce Server が実行されるサーバーマシン上の MapForce Server API から実行することができます。（マッピングを MapForce Server 実行可能ファイルへコンパイルするを参照）。

GET メソッドを使用してリアルタイムに飛行場の状態を取得するジェネリックな HTTP Web サービスの呼び出しの作成が完了しました。Web サービスとの作業に関する一般的な情報に関しては、次を参照してください

10.7 例 :RSS フィードからのデータのマッピング

この例は、RSS (Rich Site Summary) フィードからのデータをMicrosoft Excel にマップするために、MapForce からジェネリックなHTTP Web サービスを呼び出す方法について説明しています。この例は、次のパスで見つけることのできるマッピングデザインファイルが伴われます : **<Documents>\ Altova\ MapForce2017\ MapForceExamples \ RssReader.mfd** .

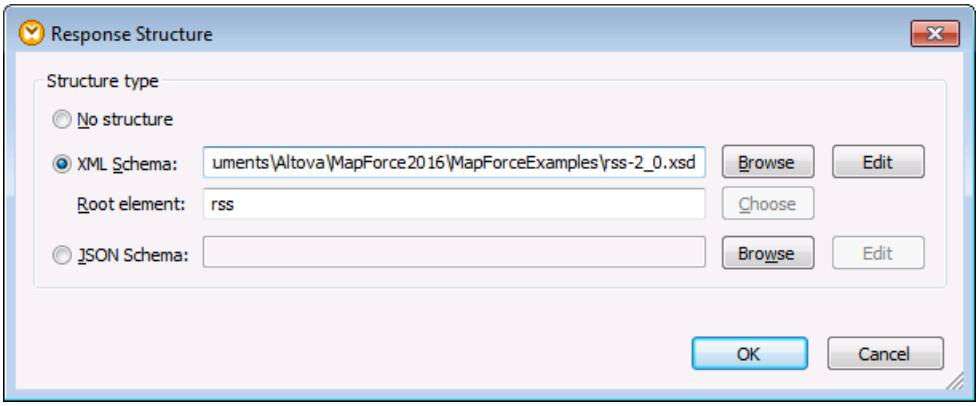
この例では、Web サービスのレスポンスがAltova ブログのRSS フィードからマップされます (http://blog.altova.com/feed)。Web サービスレスポンス構造のスキーマが必要になり、MapForce は、フィードより返されたデータの構造を作成することができます。必要なスキーマをマイドキュメント フォルダーに相対的な次のパスで見るけることができます ::**<Documents>\ Altova\ MapForce2017\ MapForceExamples\ rss-2-0.xsd**.

RSS フィードの呼び出しは、リクエストパラメーターを必要としません。ですから、 Web サービスの呼び出しのリクエストパートは空です。レスポンスは、Microsoft Excel ファイルにマップされます。また、各 RSSエントリの公開日を読みやすくするために、Excel ファイル内でYYYY-MM-DD にフォーマットされます。

ですから、マッピングは、次の3つの主要なコンポーネントから構成されます :Web サービスの呼び出し、ターゲットExcel コンポーネント、データ処理関数。

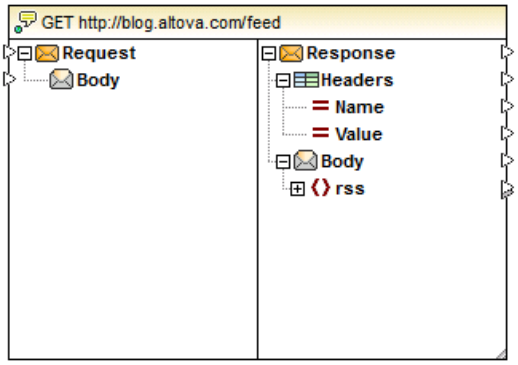
ステップ 1 : Web サービスコンポーネントを追加する

1. マッピングの変換言語がBUILT-IN に設定されていることを確認してください (変換言語の選択 を参照する)。
2. **「挿入」**メニューから、**「Web サービス関数」**をクリックします。
3. **「サービスの定義」**から **「手動」**をクリックします。
4. リクエストメソッドをGET に設定し、URL を`http://blog.altova.com/feed` に設定します。
5. **「レスポンス」**から **「編集」**ボタンをクリックして、Web サービスレスポンスのためのスキーマを参照します。スキーマは以下のパスで検索することができます :**<Documents>\ Altova\ MapForce2017 \ MapForceExamples\ rss-2-0.xsd**.



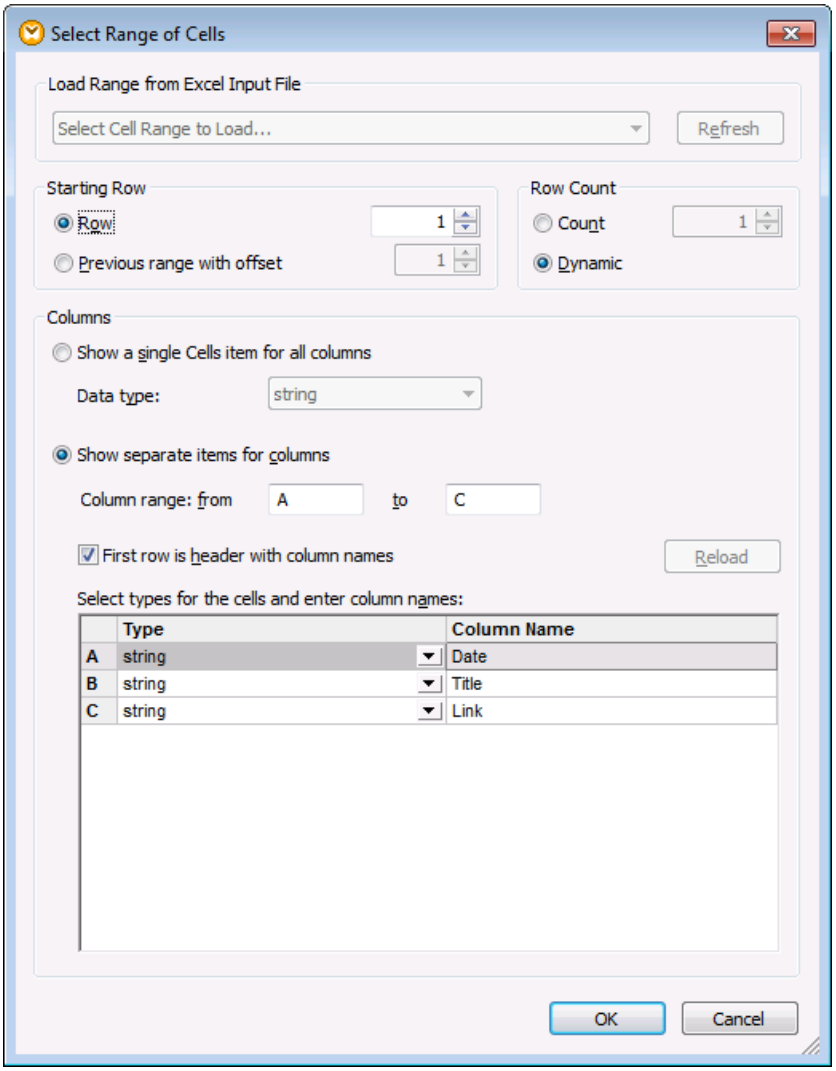
6. **「OK」**をクリックします。

この段階では、Web サービスは、マッピングエリアで使用することができ、レスポンスパートからデータをマップすることができます。



ステップ 2：Add ターゲット Excel コンポーネント

1. **挿入**」メニューから、**「Excel 2007+ ファイル**」をクリックします。 .
2. サンプルファイルを与えるようにプロンプトされると、**「スキップ**」をクリックします。Excel コンポーネントをマッピングエリアで使用することができます（サンプル JSON ファイルは生成されるため、必要ありません）。
3. **シート 1** の横のボタン をクリックします。**「名前順にワークシートを表示**」オプションのチェックを解除するためにクリックします。
4. **行 1** の横のボタン をクリックします。下に示されるようにオプションを変更します。



5. レスポンス本文の次のアイテムを Excel コンポーネントの次のアイテムに接続します :

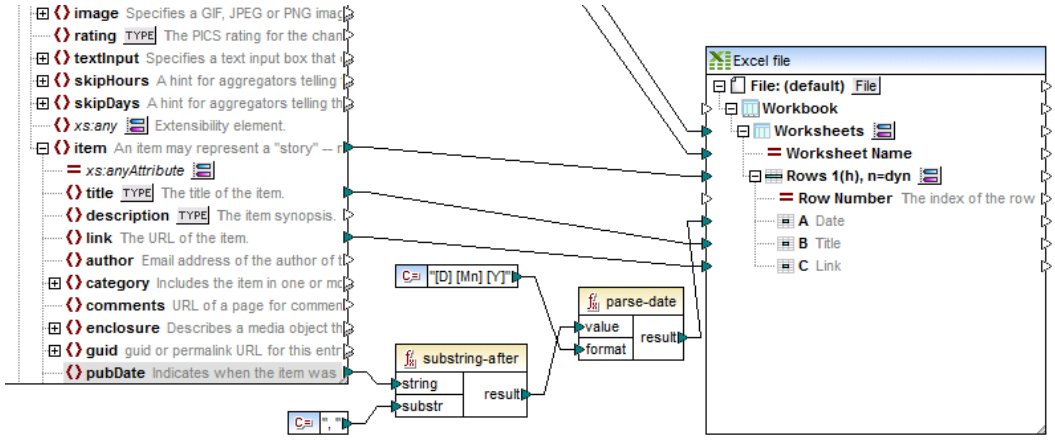
レスポンス本文	Excel ファイル
`channel`	`Worksheets`
`title`	`Worksheet Name`
`item`	`Rows`
`item/title`	`Row B (Title)`
`item/link`	`Row C (Link)`

この段階では、**行 A (Date)** の入力の値が不足しています。この値は次のステップで追加されます。

ステップ 3 : 日付処理関数を追加する

RSS フィード内のレスポンス内の日付は、RFC-822 フォーマットで表示されます。例 :`Wed, 20 Jan 2016 14:49:35 +0000`。目標は、しかしながら、`YYYY-MM-DD` として、Excel ファイル ファイル内で日付をフォーマットする

ことです。これを達成するには、MapForce core ライブラリの`substring-after` と`parse-date` 関数マッピングに追加します（関数をマッピングに追加する方法に関しては、次を参照してください：関数と作業する）。



上に表示されるとおり、`substring-after` 関数は、**pubDate** ノードから値を取り、`, ` の後のテキストを返します。結果の値は マスク`[D] [Mn] [Y]` を使用して解析し、値を`xs:date` として返す MapForce core 関数`parse-date` にパスされます。フォーマットマスクは、月初めから数えた日の数値、スペース、月名、4桁の年数の順番で表示することを意味します（詳細に関しては、次を参照してください：`format-dateTime` 関数）。

最後に、解析された値は、Excel コンポーネントの **日付** 行に与えられます。

ステップ 4：マッピングの実行

Web サービスを呼び出す準備が整うと、**出力** ボタンをクリックして、マッピングを実行し、一般的な出力をプレビューします。Web サービス呼び出しに成功すると、**出力** タブは、返されたデータ構造を表示します。呼び出しに失敗すると、MapForce は、エラーを状況に応じて返します。

MapForce Server が存在する場合、（http://www.altova.com/ja/mapforce/mapforce-server.html）マッピングをマッピング実行可能ファイル（*.mfx）にコンパイルし、コマンドライン、または、MapForce Server が実行されるサーバーマシン上の MapForce Server API から実行することができます。（次を参照：マッピングを MapForce Server 実行可能ファイルへコンパイルする）。

GET メソッドを使用してリアルタイムに RSS フィード取得するジェネリックな HTTP Web サービスの呼び出しの作成が完了しました。Web サービスとの作業に関する一般的な情報に関しては、次を参照してください：Web サービス呼び出しの追加（REST-スタイル）

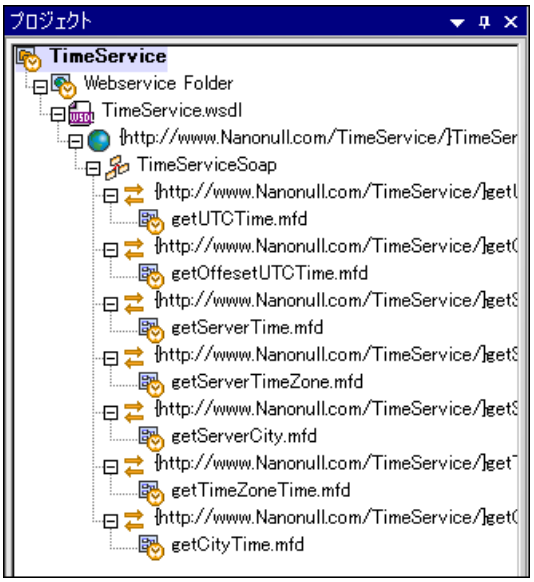
10.8 サンプル :WSDL- スタイル Web サービスの呼び出し

以下のサンプルでは、MapForce により実装された Web サービスを呼び出します。このサンプルはデモを行うために作成されたもので、互換性のあるプロトコルを使用したその他テクノロジーにより実装された Web サービスを呼び出すことも可能です。

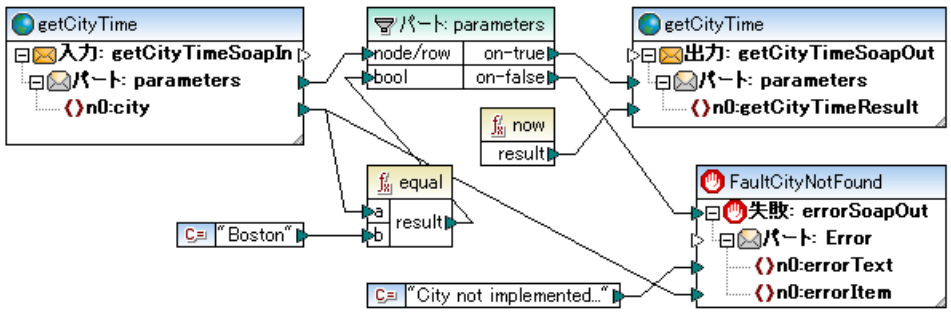
<Documents>\ Altova\ MapForce2017\ MapForceExamples\ TimeService フォルダーに含まれている**TimeService.mfp** マッピングプロジェクトの一部を以下に示します。 **<Documents>\ Altova \ MapForce2017\ MapForceExamples\ TimeserviceWsdl2** フォルダーに含まれている **TimeService2.mfp** プロジェクトファイルでは WSDL 2.0 もサポートされます。

Web サービスがどのように実装されているかをビューする:

1. **<Documents>\ Altova\ MapForce2017\ MapForceExamples\ Timeservice** フォルダーに含まれている**TimeService.mfp** ファイルを開きます。プロジェクトに関連したファイルはプロジェクトウィンドウ内にロードされます。



2. プロジェクトウィンドウ内の**getCityTime.mfd** エントリをダブルクリックします。



getCityTime.mfd マッピングは、都市の名前を入力として受け、現在の時刻を出力として返します。都市が "Boston" ではない場合、WSDL フォールトが返されます。 **<Documents>\Altova\MapForce2017 \MapForceExamples\TimeService** ディレクトリ内にある**getCityTimeRequest.xml** ファイルから入力データをマッピングは取得します。**getCityTimeSoapIn** コンポーネントをダブルクリックすると、このファイルがデータソースとして設

定されていることを確認できます。

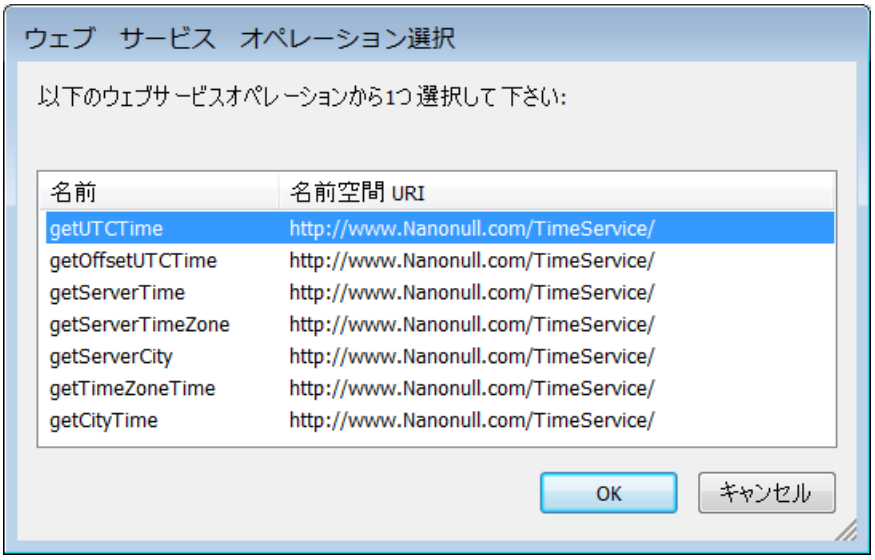
この例のスコープのために、この特定の Web サービス プロジェクトは既にコンパイルされ、このサンプルで次のステップとして呼び出される実在の Web サーバーにデプロイされていると仮定します。

"getCityTime" Web サービス関数をマッピングから呼び出す :

1. **「ファイル | 新規作成」** を選択して、**「マッピング」** アイコン をクリックし、**「OK」** をクリックします。
2. メニュー オプション **挿入 | Web サービス関数 ...」** を選択、または  ツールバー をクリックします。
3. WSDL 定義ファイルを選択するために **参照」** をクリックします。TimeService ディレクトリから **TimeService.wsdl** を選択し、**開く** ボタンクリックします。
4. Web サービス エンドポイントを選択するように問われると、**TimeServiceSoap** をクリックします。

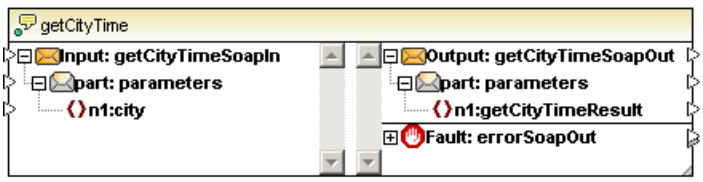


5. Web サービス操作を選択するように問われると、**getCityTime** をクリックします。



6. WSDL 呼び出し設定を指定するように問われると、設定を変更せず、　**「OK」** をクリックします。更に詳しい情報に関しては、Web サービス呼び出し設定を参照してください。

getCityTime Web サービス関数は、単一コンポーネントとして挿入されます。WSDL プロジェクト内で保存されたとおりに **getCityTime.mfd** ファイルを構成する 8つすべてのコンポーネントを示していることに注意してください。

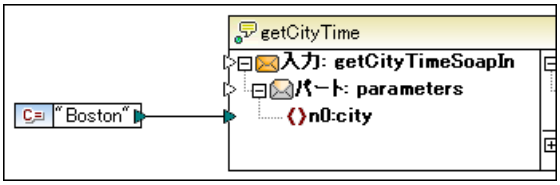


.wsdl ファイル内で、以下のうちの1つが定義されている場合、コンポーネントの左側でデータ入力 データ入力

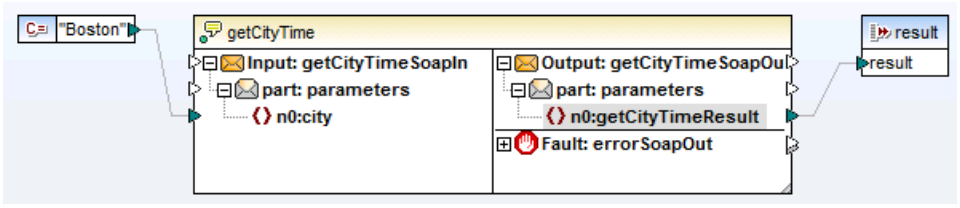
(SoapIn) が定義され、フォールトのセクションを含んでいる可能性がある右側でデータ出力 (SoapOut) が定義されます。

Web サービス関数のための入力と出力コンポーネントを定義する:

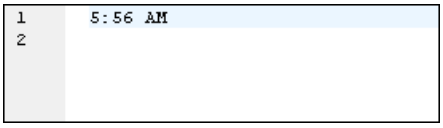
1. たとえば、定数、テキスト、またはスキーマコンポーネント、入力データを提供するためのコンポーネントを挿入します。この例のスコープのために、定数コンポーネントを挿入して、"Boston" を入力文字として挿入します。



2. 定数を**n1:city** アイテムに接続します。
3. 単純型出力コンポーネントを挿入します (**関数** メニューから、**出力の挿入**) をクリックします)。
4. n1:getCityTimeResult を出力コンポーネントに接続します。



5. マッピングの出力を確認するために出力タブをクリックします。

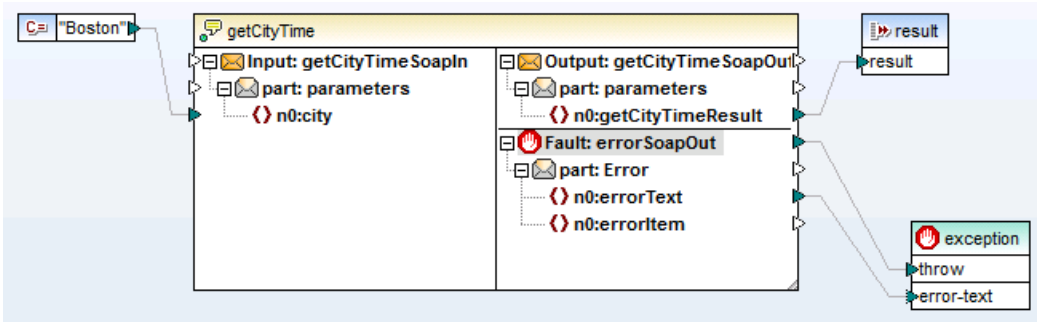


Boston の現在の時刻が出力ウィンドウに表示されます .

メモ: Web サービス関数の入力値は、元のマッピングのデータソースを優先します。t例えば、定数コンポーネントの値 "Boston" は、元のマッピング内の**getCityTimeRequest.xml** データソースファイルを優先します。

Web サービスのマップに失敗した場合 :

1. **挿入 | 例外** を選択、または ツールバーボタン をクリックします。
2. **Fault:** アイテムを例外コンポーネントの**throw** アイテムにマップします。
3. **n1:errorText アイテムを** 例外コンポーネントの **error-text** アイテムにマップします。



10.9 Digital Certificate Management

Digital certificate management is an integral part of secure data exchange between a client computer and a Web server. Since mappings can be executed not only on Windows by MapForce, but also on a Windows, Linux or Mac server by MapForce Server (either standalone or in FlowForce Server execution), this section deals with managing HTTPS certificates on various platforms.

In the context of secure HyperText Transport Protocol (HTTPS), it is important to distinguish between server and client certificates.

Server certificates

A server certificate is what identifies a server as a trusted entity to a client application such as MapForce. The server certificate may be digitally signed by a commercial Certificate Authority, or it may be self-signed by your organization. In either case, while designing the mapping in MapForce, you can specify the following settings:

- Whether the server certificate must be checked.
- Whether the request must proceed if a mismatch has been detected between the name certificate and the name of the host.

These settings are available on the HTTP Security Settings dialog box of MapForce (see Setting HTTP Security). When you enable server certificate checks, consider the following:

- If you are calling a Web server whose certificate is signed by a trusted Certificate Authority, your operating system will likely be already configured to trust the server certificate, and no additional configuration is necessary.
- If you are calling a Web server which provides a self-signed certificate (for example, a local network server within your organization), you will need to configure your operating system as well to trust that certificate.

In most cases, you can check the level of trust between your operating system and the Web server by typing the URL of the Web service in the browser's address bar. If the server is not trusted, or if your operating system is not configured to trust the server, your browser will display a message such as "This connection is untrusted", or "There is a problem with this website's certificate". Note that you cannot use the browser to check the level of trust with a Web server if the browser uses a certificate database other than that of the operating system (for example, Firefox 35.0.1 on Ubuntu 14.04).

On Windows, you can establish trust with the server by following the browser's instructions and importing or installing the required certificates into your system's Trusted Root Authorities store (see Trusting Server Certificates on Windows). On Mac, you can do the equivalent operation in Safari (see Trusting Server Certificates on Mac). For instructions applicable to Linux, see Trusting Server Certificates on Linux.

Client certificates

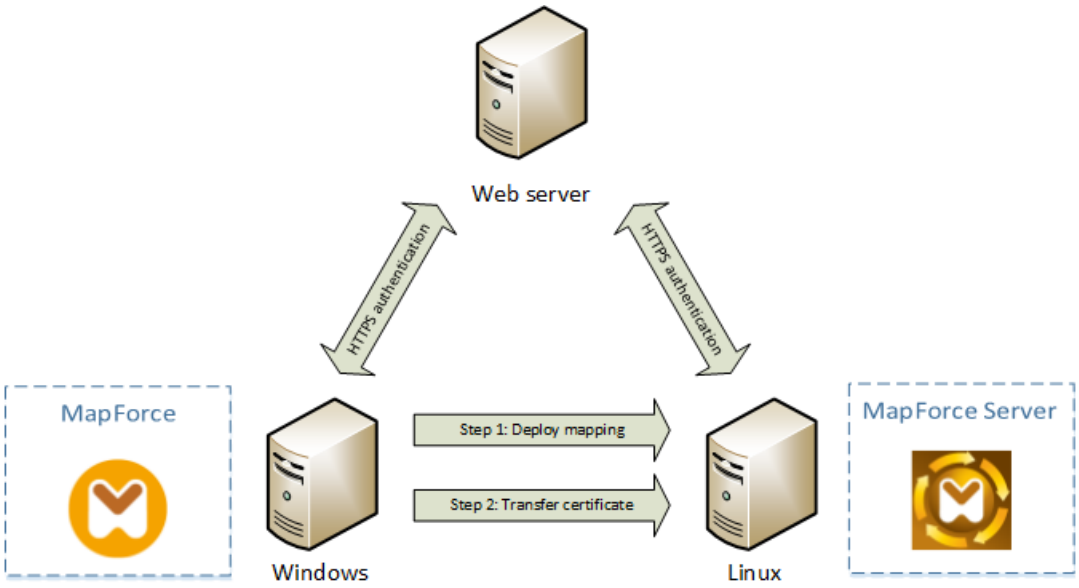
While server certificates are used to identify a server as a trusted entity, client certificates are primarily used to authenticate the caller against the Web server. If you intend to call a Web server which requires client certificates, you may need to contact the administrator of the Web server for the client configuration instructions. Taking IIS (Internet Information Services) as an example, the Web server may be configured to handle HTTPS and client certificates in one of the following ways:

- Require HTTPS and ignore client certificate
- Require HTTPS and accept client certificate
- Require HTTPS and require client certificate

The success or failure of the Web service request depends both on the configuration of the Web server and the client application. For example, if the Web server is configured to require a client certificate, then, for the call to be successful, the calling application must present a valid client certificate.

From a MapForce perspective, the same is true for mappings which include Web service calls through HTTPS. In particular, to run such mappings successfully, it is assumed that the Web server has been configured to accept or require the client certificate, and that the operating system where the mapping runs provides the correct client certificate to the Web server.

The diagram below illustrates a scenario where a client certificate used in MapForce is transferred to a Linux server running MapForce Server. Once the certificate has been transferred to the target operating system, MapForce Server can use it to authenticate itself against the Web server and execute the mapping successfully.



Deploying mappings with client certificates to another computer

For HTTPS authentication in Web service calls, MapForce is capable of using Transport Layer Security (TLS) on top of HTTP, which is the successor of Secure Sockets Layer (SSL) protocol. Note that fallback to SSL may occur if either the client implementation or the server does not support TLS.

To support Web calls with client certificate authentication on multiple platforms, MapForce (and MapForce Server) relies on the certificate management implementation of each platform, thus ensuring that certificate management is always in the scope of the underlying operating system. Each operating system provides different support for certificate management, as shown in the table below.

Platform	Certificate management and implementation
Windows	On Windows, you can manage certificates using the Certificate snap-in (see Accessing the Certificate Stores on Windows). TLS support is available through the *Secure Channel* (also known as *SChannel*) library.
Linux	On Linux, you can manage certificates using the OpenSSL (`openssl`) command line tool and library. If OpenSSL support is not already available on the Linux machine where MapForce Server is installed, you will need to download and install it before you can manage certificates. TLS support is available through the OpenSSL library (https://www.openssl.org/).
Mac	On Mac, you can manage certificates using the *Keychain Access Manager*, located under **Finder > Applications > Utilities**. TLS support is provided by the *Secure Transport* library native to the operating system.

If you execute the mapping on a Windows operating system where you can already successfully consume the same Web service that you intend to call from MapForce, no additional certificate configuration is normally required (for the conditions to run the mapping successfully on Windows, see Client Certificates on Windows). However, if you design mappings with MapForce on a Windows computer, and then deploy them to another computer (which may run a different operating system), the client certificate is not stored or copied together with the deployed package. For the Web service call (and the mapping) to execute successfully, the client certificate must exist on the target operating system as well.

To transfer a certificate from a Windows system to another Windows-based computer, export the required certificate (with private key) from the source system (see Exporting Certificates from Windows). Then import the same certificate to the **Current User\Personal** store on the target operation system (see Client Certificates on Windows).

For instructions on how to transfer client certificates to the Linux and Mac platforms, see Client Certificates on Linux and Client Certificates on Mac, respectively.

10.9.1 Trusting Server Certificates on Linux

To establish trust with a Web server on Linux, obtain the certificate file of the Web server, copy it to the system's certificate store, and then update the latter (see instructions below). One of the ways to obtain the server certificate is by using the Firefox browser, as shown in the example below.

> Perform the following steps only if you are sure of the authenticity of the Web server certificate.

Debian, Ubuntu

1. Copy the certificate file of the Web server to the following directory.

```
sudo cp /home/downloads/server_cert.crt /usr/local/share/ca-certificates/
```

2. Update the certificate store as follows:

```
sudo update-ca-certificates
```

Cent OS

1. Install the `ca-certificates` package:

```
yum install ca-certificates
```

2. Enable the dynamic certificate authority configuration feature:

```
update-ca-trust enable
```

3. Copy the server certificate to the following directory:

```
cp server_cert.crt /etc/pki/ca-trust/source/anchors/
```

4. Use the command:

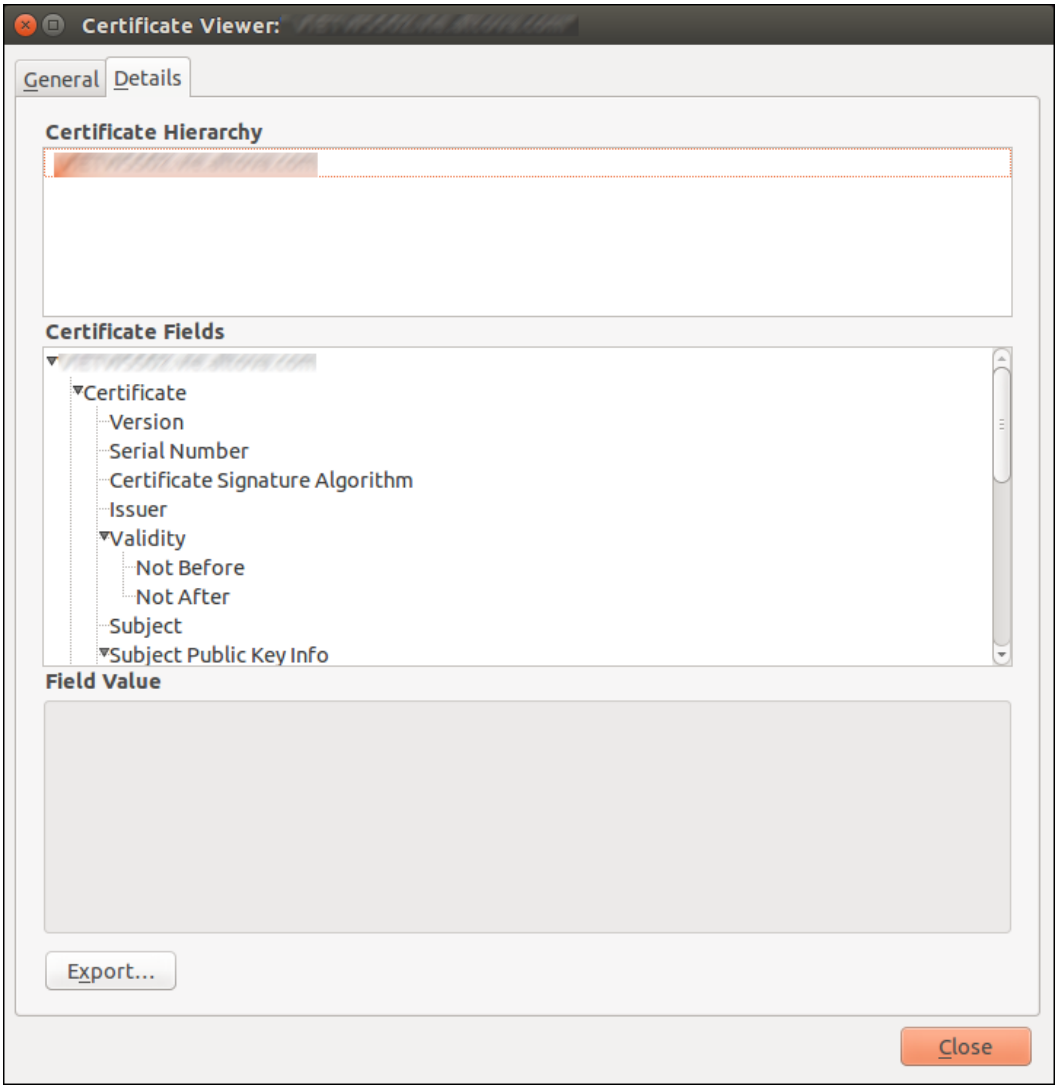
```
update-ca-trust extract
```

Example: Exporting the certificate of the Web server with Firefox on Ubuntu

1. Run Firefox and access the URL of the Web server.
2. When prompted with the message "This connection is untrusted", click **Add Security Exception**.



3. Click **View**.
4. Click the **Details** tab.



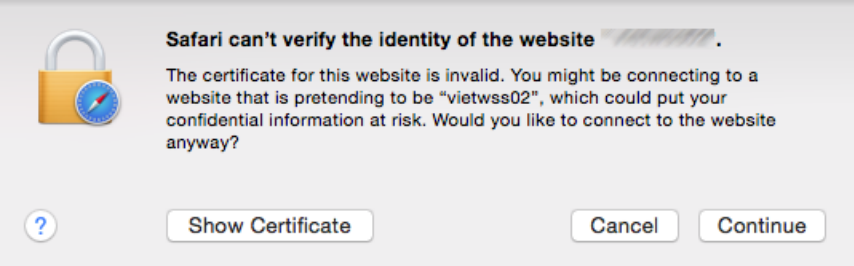
5. Click **Export** and save the certificate file to a local directory.

10.9.2 Trusting Server Certificates on Mac

To establish trust with a Web server using Safari:

Perform the following steps only if you are sure of the authenticity of the Web server certificate.

1. In the browser address bar, enter the HTTPS address of the Web server.
2. When prompted to connect to the website, click **Show Certificate**.



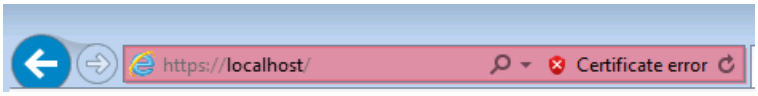
3. Select the option **Always trust {certificate} when connecting to {website}**.
4. Click **Continue** and enter your password when prompted.

10.9.3 Trusting Server Certificates on Windows

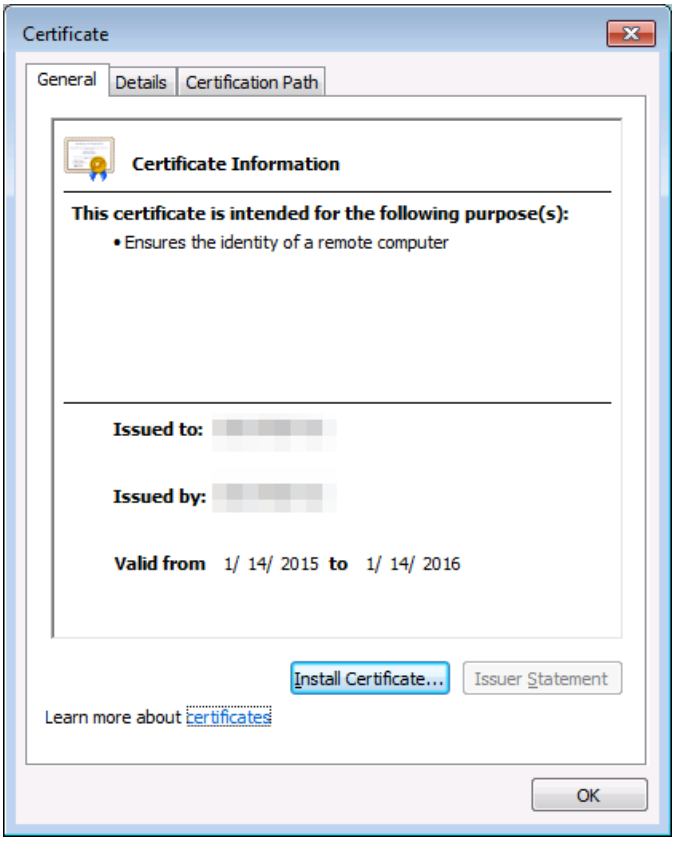
To establish trust with a Web server using Internet Explorer 11:

> Perform the following steps only if you are sure of the authenticity of the Web server certificate.

1. In the browser address bar, enter the HTTPS address of the Web server.
2. Click **Continue to this website (not recommended)**.
3. Click the Certificate error area, and then click **View certificates**.



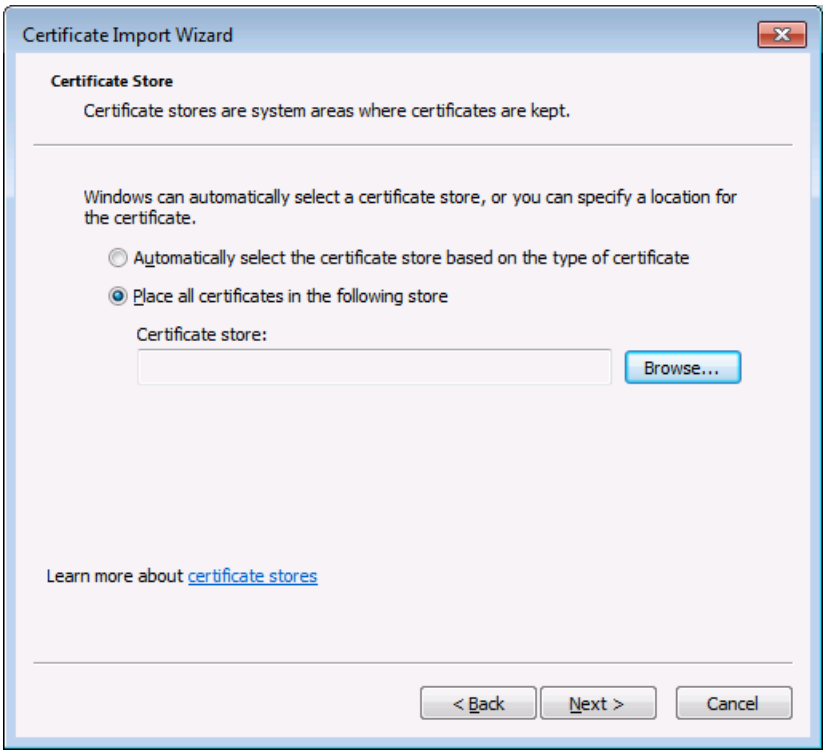
4. Click **Install Certificate**.


Certificate
General
Details
Certification Path
Certificate Information
This certificate is intended for the following purpose(s):
• Ensures the identity of a remote computer
Issued to:
Issued by:
Valid from 1/ 14/ 2015 to 1/ 14/ 2016
Install Certificate...
Issuer Statement
Learn more about certificates
OK

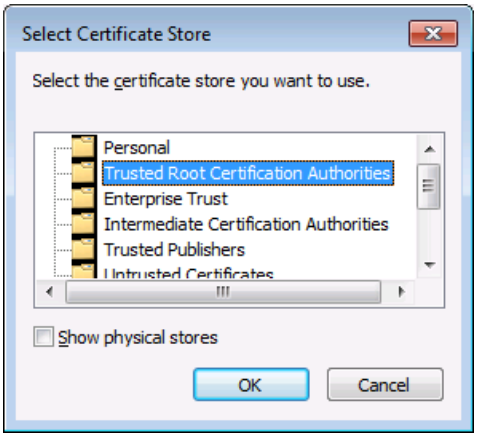
5. Click **Next**.


Certificate Import Wizard
Welcome to the Certificate Import Wizard
This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.
A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.
To continue, click Next.
< Back
Next >
Cancel

6. Choose to select a store manually.



7. Browse for the Trusted Root Certification Authorities, and then click OK.



8. When prompted to confirm your action, click OK.

10.9.4 Accessing the Certificate Stores on Windows

On Windows, you can manage certificates either from the *Certificates* Microsoft Management Console (MMC) snap-in, or from Internet Explorer.

To open the Certificates snap-in (for the current Windows user):

- Run `certmgr.msc` at the command line.

To open the certificate management dialog box in Internet Explorer:

1. On the **Tools** menu, click **Internet Options**.
2. Click the **Content** tab, and then click **Certificates**.

10.9.5 Exporting Certificates from Windows

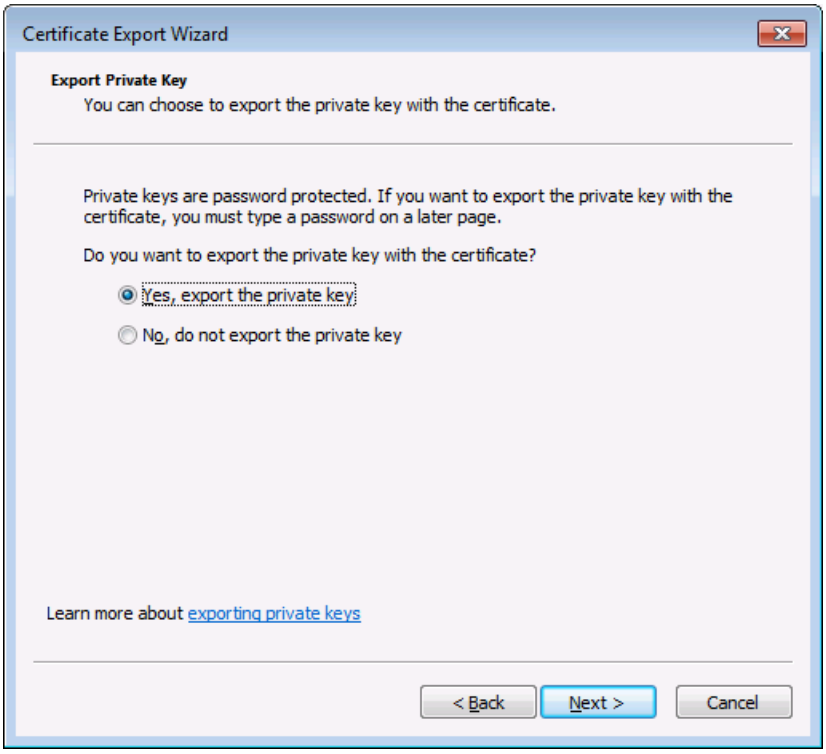
For mappings that call Web services through HTTPS and are deployed to a Mac or Linux server running MapForce Server or FlowForce Server, the same client certificate must be available on the non-Windows operating system as the one used on Windows to design and test the mapping. To execute such mappings on a non-Windows operating system with MapForce Server, export the required certificate with private key from Windows and then import it into the target operating system.

To export a certificate with private key from Windows:

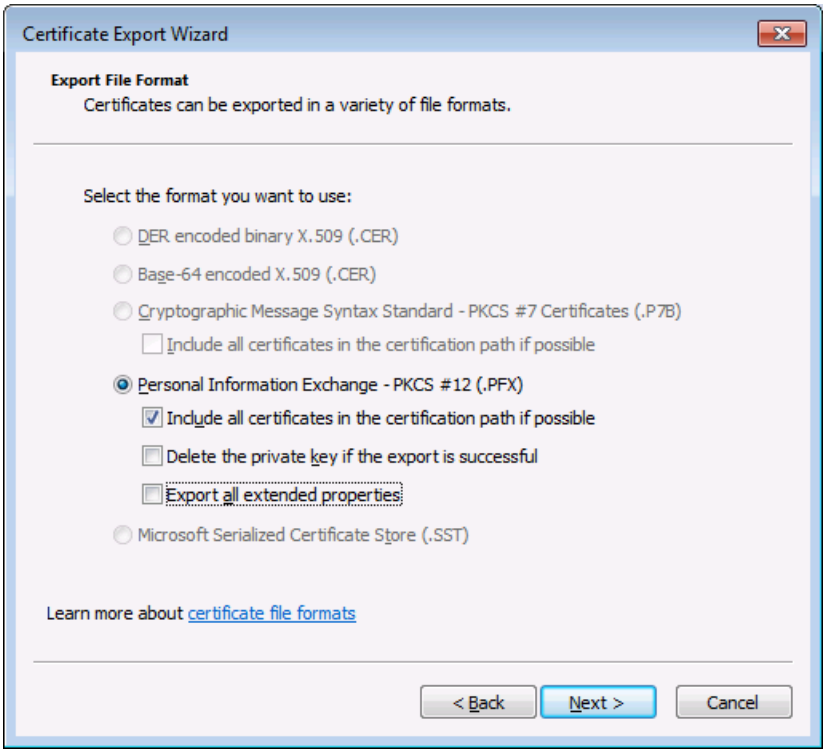
1. On Windows, open the Certificates snap-in (see Accessing the Certificate Stores on Windows).
2. Right-click the certificate that you want to export, point to **All Tasks**, and then click **Export**.
3. Click **Next**.



4. Choose to export from Windows the certificate together with its private key, and then click **Next**.

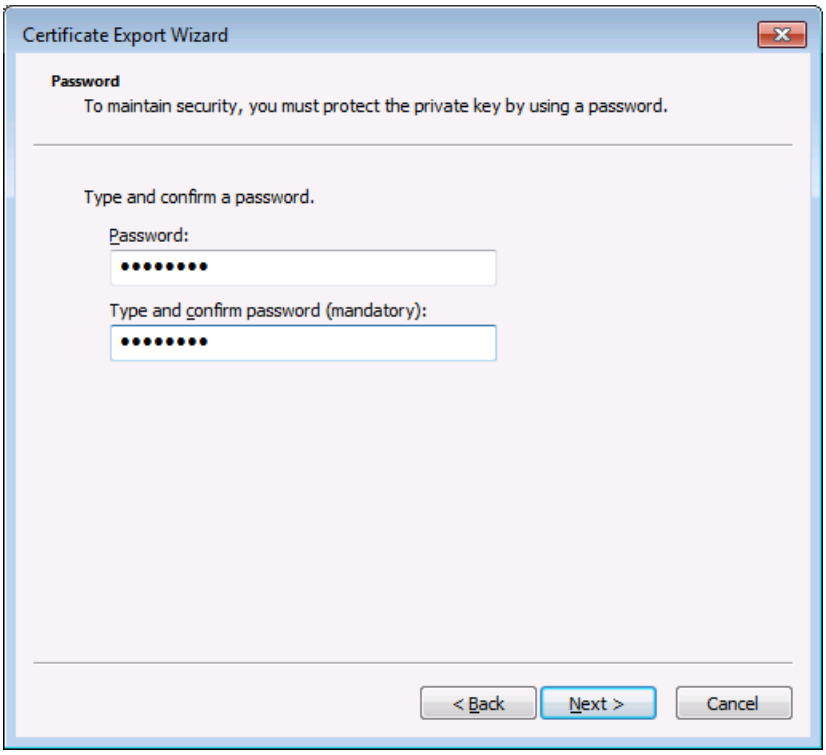


5. Choose the *Personal Information Exchange - PKCS #12 (.pfx)* file format, and then click **Next**.

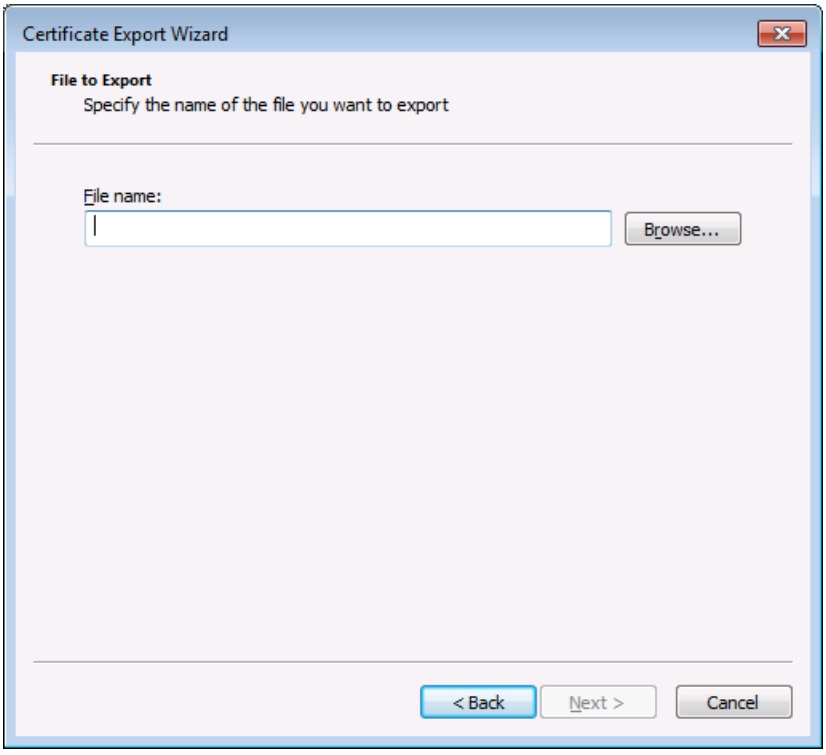


Note: Make sure not to select the option **Delete the private key if the export is successful**, otherwise you will not be able to make use of the certificate after it is exported.

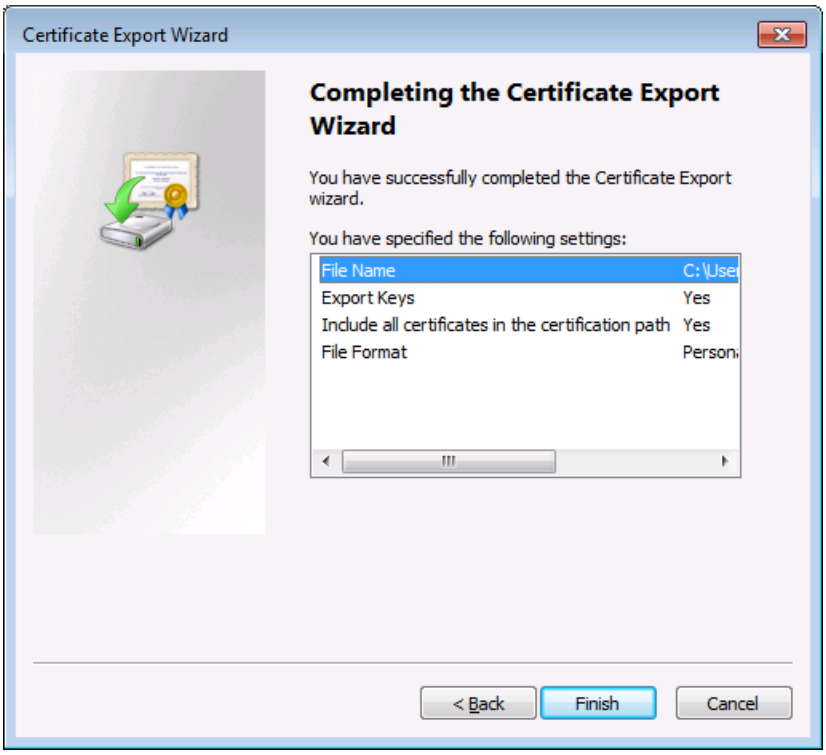
6. Enter a password, and then click **Next**. You will need this password after you copy the certificate to the target operating system.



7. Browse for the location of the file to export, and then click **Next**.



8. Click **Finish**.



10.9.6 Client Certificates on Linux

If your mappings include Web service authentication through HTTPS by means of client certificates, follow these steps to deploy such mappings to a Linux machine running MapForce Server:

1. In the HTTP Security Settings dialog box, click **Client Certificate**, and then select the required certificate from the **Current User\Personal** store on Windows (see Setting HTTP Security).
2. Save and deploy the mapping to the target operating system (see Deploying a MapForce mapping).
3. Transfer the client certificate required by the Web service call to the target operating system. Make sure that the certificate has a private key, and that the **Enhanced Key Usage** property of the certificate includes "Client authentication" as purpose.

To transfer the client certificate to Linux:

1. Export the client certificate with private key from Windows, in the *Personal Information Exchange - PKCS #12 (.pfx)* file format (see Exporting Certificates from Windows).
2. Copy the certificate file to the Linux machine.
3. Convert the .pfx file to .pem format using the command:

```
openssl pkcs12 -in cert.pfx -out "John Doe.pem" -nodes
```

This command parses the .pfx file and outputs a .pem file, without encrypting the private key. Certificates with an encrypted private key prompt for password and are not supported

in server execution.

Executing the mapping

To instruct MapForce Server to use the .pem file as client certificate, set the `--certificatespath` parameter when running the mapping. The `--certificatespath` parameter defines the path of the directory where all certificates required by the current mapping are stored. For example, if the certificate file path is `/home/John/John Doe.pem`, then `--certificatespath` must be set to `/home/John`.

By default, if the `--certificatespath` parameter is not provided, MapForce Server looks for certificates in the directory `$HOME/.config/altova/certificates` of the current user.

> For the mapping to execute successfully, the certificate file is expected to have the .pem extension and the file name must match the Common Name (CN) of the certificate, including spaces (for example, **John Doe.pem**). If the CN contains a forward slash (/), it must be replaced with an underscore (_) character.

If you intend to execute the mapping as a FlowForce Server job, copy the certificate file to the `$HOME/.config/altova/certificates` directory. When running the job, FlowForce Server will use this directory to look for any certificate files required by the mapping.

> For security considerations, make sure that certificate files are not readable by other users, since they contain sensitive information.

10.9.7 Client Certificates on Mac

If your mappings include Web service authentication through HTTPS client certificates, follow these steps to deploy such mappings to a OS X running MapForce Server:

1. In the HTTP Security Settings dialog box of MapForce, click **Client Certificate**, and then select the required certificate (see Setting HTTP Security).
2. If the certificate name does not match exactly the host name of the server, select **Allow name mismatch between certificate and request**.
3. Save and deploy the mapping to the target operating system (see Deploying a MapForce mapping).
4. Transfer the client certificate required by the Web service call to the target operating system. Make sure that the certificate has a private key, and that the **Enhanced Key Usage** property of the certificate includes "Client authentication" as purpose.

To transfer the client certificate to Mac:

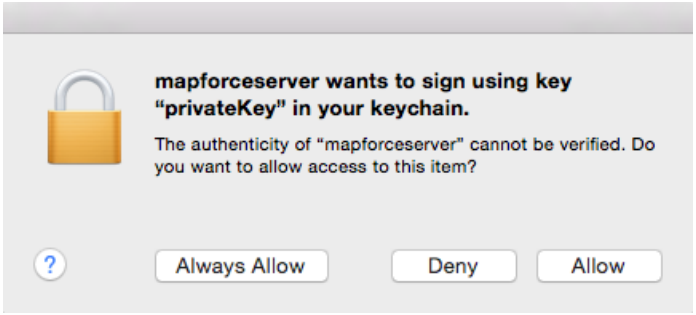
1. Export the client certificate with private key from Windows, in the *Personal Information Exchange - PKCS #12 (.pfx)* file format (see Exporting Certificates from Windows) and copy the .pfx file to the Mac.
2. If this hasn't been done already, make sure that the operating system trusts the server certificate (see Trusting Server Certificates on Mac OS).
3. Run Keychain Access from **Finder > Applications > Utilities**.
4. On the **File** menu, click **Import Items**.
5. Browse for the the client certificate exported from Windows in step 1 and select a destination keychain.

6. Click **Open** and enter the password with which the certificate was encrypted.

Executing the mapping

You are now ready to run the mapping using the MapForce Server `run` command. Note the following:

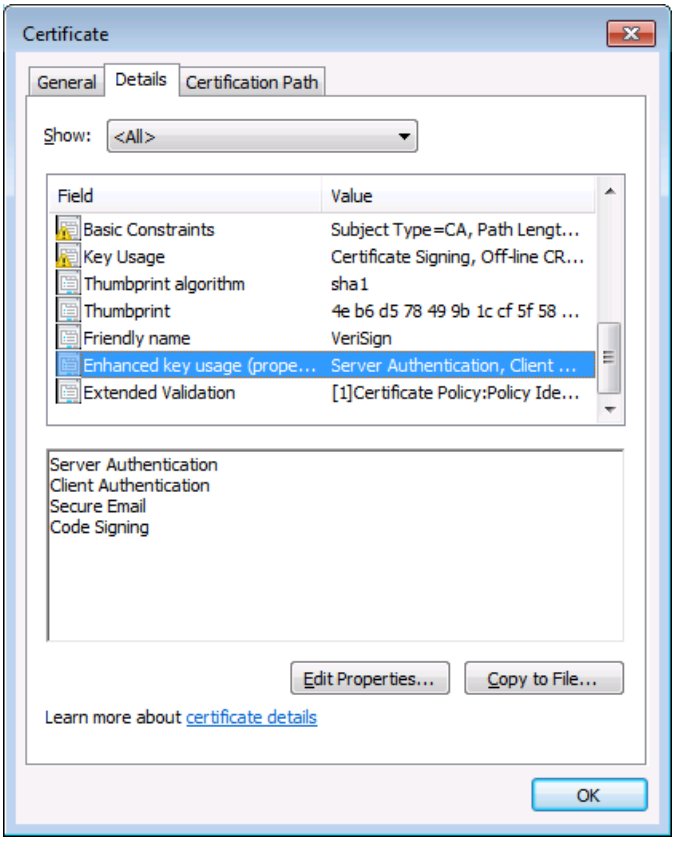
- If you execute the mapping remotely through SSH, first unlock the keychain with the `security unlock-keychain` command.
- If you execute the mapping through the Mac OS graphical user interface, when prompted to allow MapForce Server access to the keychain, click **Allow**.



10.9.8 Client Certificates on Windows

When you run on Windows a mapping which requires client certificates, the conditions to run the mapping successfully are as follows:

- The client certificate must exist in the **Current User\Personal** certificate store (also referred to as the **My** store). For the certificate to exist in this store, it must be imported through the Certificate Import Wizard. For instructions, see http://windows.microsoft.com/en-us/windows/import-export-certificates-private-keys#1TC=windows-7.
- The certificate must have a private key.
- The **Enhanced Key Usage** property of the certificate must include "Client authentication" as purpose.



In the current version of MapForce, due to a limitation of the library used by MapForce, Windows will select the required certificate automatically from the certificate store when you run the mapping. The mapping will execute successfully if, after filtering the **Current User\Personal** certificate store, the server finds a suitable certificate. Note that the HTTPS authentication (and the certificate selection operation) is managed by Windows and is not controlled by MapForce or MapForce Server. In some cases, if multiple certificates exist in the **Current User\Personal** store, an unsuitable certificate may be selected automatically by the operating system, which causes the mapping execution to fail. This situation can be avoided by limiting the number of certificates available in the **Current User\Personal** store.

Chapter 11

MapForce とマッピングの自動化

11 MapForce とマッピングの自動化

このセクションは、のコマンドラインインターフェイスおよびAltova サーバーツール:RaptorXML Server、MapForce Server、およびFlowForce Serverを使用してマッピングを自動化する方法について説明します。

11.1 MapForce Server について

MapForce Server は、Windows、Linux とMac OS X オペレーティングシステムのためのエンタープライズサーバーソフトウェアソリューションです。MapForce Server の役割は、(非 Windows プラットフォームを含む)サーバー環境で、サーバーレベルのマッピングを実行することです。MapForce マッピング内でBUILT-IN がターゲット実行言語である箇所でサーバー実行をおこなうことができます (次も参照 変換言語の選択)。MapForce Server は、(コマンドライン、または、API から呼び出される)スタンドアロンで、または、FlowForce Server の管理下で作動します。

MapForce Server がスタンドアロン製品として使用されている場合、MapForce マッピングは、コンパイルされ MapForce Server が実行されているマシンにコピーされる必要があります。MapForce Server コマンドラインコマンド `run` を使用してマッピングは実行されます。MapForce Server API の `run` メソッドを呼び出してマッピングを呼び出すこともできます。詳しい情報に関しては、以下を参照してください：MapForce Server 実行ファイルへマッピングをコンパイルする。

MapForce Server がFlowForce Server 管理下で実行されている場合、マッピングをHTTP (またはSSL/HTTPS) 接続を使用してMapForce から、直接ターゲットマシンにデプロイすることができます。サーバー上では、マッピングは、トリガーされた、または、予定されたジョブとして実行されます。または、FlowForce Server 管理インターフェイスから定義されたサービス呼び出しを介して実行されることができます。詳しい情報に関しては、以下を参照してください：FlowForce Server にマッピングをデプロイする。

MapForce Server には以下の2つのエディションがあります：

- MapForce Server
- MapForce Server Accelerator Edition

MapForce Server Accelerator Edition にはMapForce Server と同じ機能が搭載されており、追加して、最適化を行うことのできるマッピングに対して、マッピングを最適化する機能が含まれています。これは、多量のデータをジョイン、または、フィルターするマッピングの場合、または、多量のデータをフィルターする場合、実行スピードを向上するためにジョインの最適化が適用されます。
制約事項：

- XML 署名はサポートされていません
- COM インターフェイスを介してグローバルリソースはサポートされていません
- ODBC とADO 接続はWindows でのみサポートされており (その他のオペレーティング システムに関しては Linux とMac 上でのデータベース接続を参照してください)。

11.2 MapForce Server 実行可能ファイルにマッピングをコンパイルする

MapForce 内で作成されるマッピングのターゲット言語がBUILT-IN に設定されている場合 MapForce のみでなく MapForce Server もがマッピングを実行することができます (次を参照してください:MapForce Server について)。 MapForce Server を使用して、マッピングを実行するには、マッピングは以下の通り準備される必要があります:

- MapForce Server がスタンドアロンモードで作動している場合、 (つまり、FlowForce Server がインストールされていない状態で)、マッピングは、下に表示されるとおり、サーバー実行ファイル (.mfx) にコンパイルされる必要があります。その後、.mfx ファイルをコマンドラインでコマンド `run` を使用して実行します。`run` メソッドを MapForce Server API から呼び出して、マッピングを実行することもできます。詳細に関しては、次を参照してください:MapForce Server ドキュメント (http://manual.altova.com/MapForceServer/)。
- または、MapForce Server が FlowForce Server 管理下で作動している場合、マッピングは、MapForce Server とFlowForce Server の双方が作動しているマシンにデプロイすることができます (次を参照してください:マッピングをFlowForce Server にデプロイする)。

MapForce Server Execution (.mfx) ファイルにマッピングをコンパイルする:

1. MapForce 内でマッピングを開きます (例えば、**myMapping.mfd**)。
2. **「ファイル」** メニューから、**「MapForce Server 実行ファイルにコンパイル」** をクリックします。
3. .mfx ファイルを保存するフォルダーを選択し、必要であれば名前を変更します。
4. **「保存」** をクリックします。MapForce Server 実行ファイル **myMapping.mfx** は、選択されたフォルダー内に生成されます。

MapForce Server 実行可能ファイル (.mfx) ファイルにマッピングをコマンドラインを使用してコンパイルする:

- コマンドラインで MapForce を実行し、マッピングファイルと `/COMPILE` コマンドラインオプションを指定します。

例えば、次のコマンドは、マッピング **<Documents>\Altova\MapForce2017\MapForceExamples\SimpleTotal.mfd** を MapForce Server 実行ファイルにコンパイルし、ターゲット出力ディレクトリ **C:\Users\altova\Desktop** に作成されます。

```
"C:\Program Files (x86)\Altova\MapForce2017\MapForce.exe" "C:\Users\altova
\Documents\Altova\MapForce2017\MapForceExamples" /COMPILE "C:\Users\altova
\Desktop"
```

以下も参照してください:MapForce コマンドラインインターフェイス。

特定の MapForce Server バージョンのためにマッピングをコンパイルする

MapForce Server が MapForce よりも古いバージョンの場合、前者は、新機能が搭載されていない可能性があるため、新しいバージョンの MapForce で作成された .mfx ファイルを実行できない可能性があります。 このような場合には、.mfx ファイルを特定の MapForce Server バージョンのために、以下の方法でコンパイルすることができます:

1. **「ツール」** メニューから、**「オプション」** をクリックして、**「生成」** をクリックします。
2. **「サーバー実行ファイル」** の下の **「MapForce Server バージョンの生成」** 横から、必要とする MapForce Server バージョンをドロップダウンリストから選択します。



新しいMapForce Server バージョンを取得すると、このオプションを必要に応じて変更してください。MapForce Server の特定のバージョンのためにコンパイルする特定の理由が存在しない場合は、 '最新 "オプションを選択します (これはデフォルトのオプションです)。このオプションが選択されていると、mfx ファイルは、MapForce Server の最新のバージョンのためにコンパイルされるため、前のバージョンでは使用することのできない最新の機能を活用することができます。

ターゲットMapForce Server バージョンをコマンドラインで指定するには、`/MFXVERSION` スイッチを利用して `/COMPILE` コマンドを実行します。例：

```
"C:\Program Files (x86)\Altova\MapForce2017\MapForce.exe" /COMPILE /
MFXVERSION:2017
```

以下も参照してください：MapForce コマンドラインインターフェイス。

他のオプション

MapForce Server 実行可能ファイルのコンパイルは、次のオプションによっても影響されます：

すべてのADO とODBC データベースの接続を JDBC に変換する	オプションが有効化されていると、ADO とODBC データベースの接続は、JDBC ドライバーとデータベース コンポーネント設定ダイアログボックス内で定義されたデータベースURLを使用して、JDBC に変換されます (次を参照してください：データベースコンポーネントの設定)。 .mfx ファイルをLinux またはMac サーバーで実行する場合、このオプションを有効化する必要があるかもしれません (次を参照してください：Linux とMac 上でのデータベースの接続)。
(MapForce Server によりサポートされていない)デジタル署名を無視する	このオプションはMapForce Enterprise にのみ適用することができます。また、デフォルトで有効化されています。マッピングがXML デジタル署名を使用する場合、MapForce Server がXML デジタル署名をサポートしないため、デジタル署名に関する情報をスキップします。

これらのオプションをビュー、または、変更することができます：

- **「ツール」** メニューから、 **「オプション」** をクリックして、 **「生成」** をクリックします。

これらのオプションはコマンドラインインターフェイスからも使用することができます。以下も参照してください：MapForce コマンドラインインターフェイス

11.3 マッピングを FlowForce Server にデプロイ

FlowForce Server にマッピングをデプロイすることは、が特定のマッピングにより使用されたリソースをオブジェクトに整理し、HTTP を使用して FlowForce Server が実行されているマシンにパスすることを意味します。MapForce マッピングは、を使用して実行を自動化するために通常 FlowForce Server にデプロイされます。

FlowForce Server には MapForce マッピングがデザインされたオペレーティングシステムと同じオペレーティングシステムで実行される必要はなく、マッピングをデプロイする際、以下に注意してください：

- FlowForce Server が実行されるオペレーティングシステムの構成はマッピングの実行を許可する必要があります。例：特定のデータベースドライバーを必要とするマッピングがデータベースコンポーネントを含む場合、このようなドライバーはターゲットサーバー上に存在する必要があります。
- マッピングを非 Windows プラットフォームにデプロイする場合、ADO と ODBC データベース接続は JDBC に変更されます。詳細に関しては、データベースへ接続するを参照してください。
- マッピングに例：dll または class ファイルなどへのカスタム関数の呼び出しが含まれる場合、、このような依存関係は、ランタイムの前に既知ではないため、マッピングと共にデプロイすることはできません。この場合、手動でターゲットサーバーにコピーすることができます。

デプロイ後、マッピングは、FlowForce Server 管理インターフェイスで、指定されたパスの場所で、マッピング関数 (**.mapping**) として使用することができます。ソースコンポーネントは入力引数になり、ターゲットコンポーネントは、この関数の出力引数になります。



25

FlowForce Server 内のサンプルマッピング機能

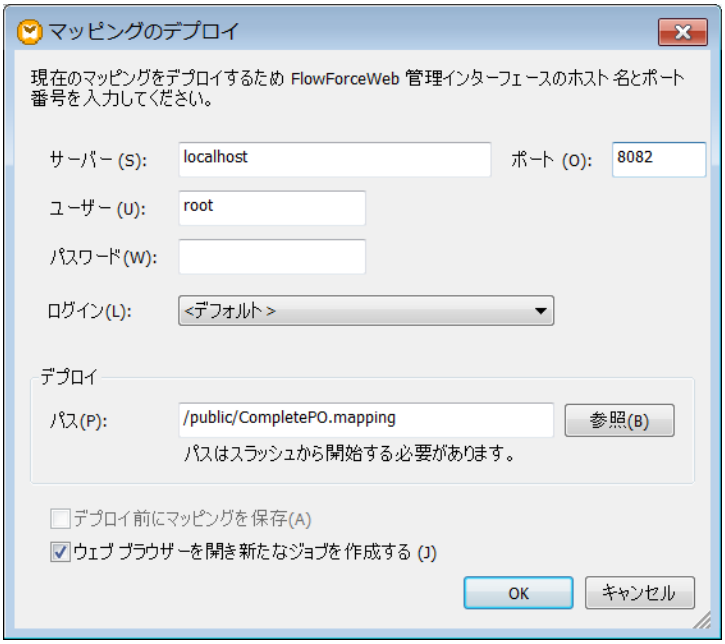
この時点から、デプロイされたマッピングから全ての機能を搭載した FlowForce Server ジョブを作成することができ、ジョブ特有の機能を活用することができます（例：ジョブのためのカスタム化されたトリガーの条件、ジョブをとして公開するなど）。

FlowForce Server ジョブとして MapForce マッピングをデプロイと実行する必要条件は、以下のとおりです：

- 必須のライセンス:MapForce Enterprise または Professional エディション、MapForce Server または MapForce Server Accelerator Edition 、FlowForce Server
- 構成されたネットワークアドレスとポートで FlowForce Server が実行されています。
- コンテナーへのパーミッションつきの FlowForce Server ユーザーアカウント。 (デフォルトでは、/public コンテナーに認証された全てのユーザーによりアクセスすることができます。)

FlowForce Server に MapForce マッピングをデプロイする:

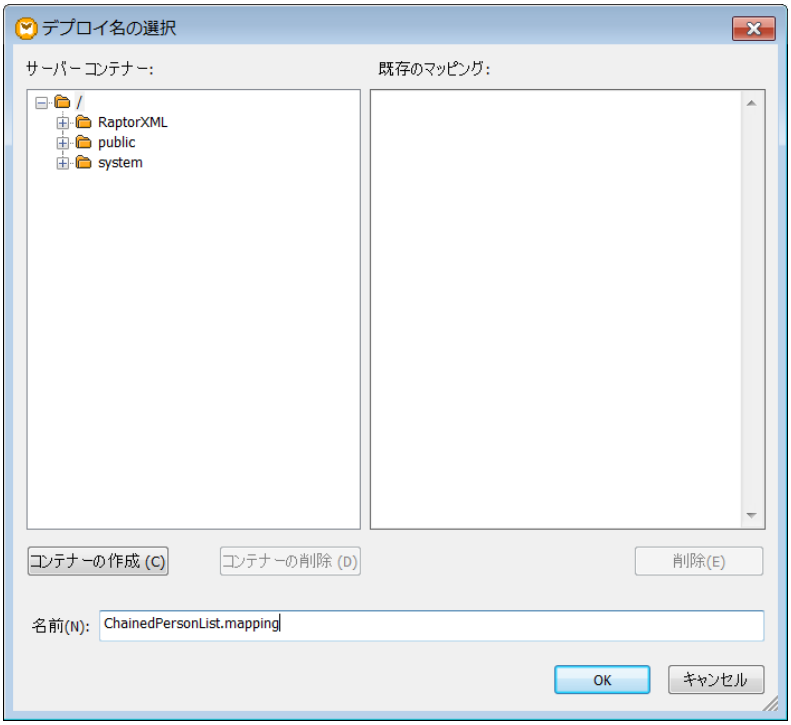
1. 変換言語が BUILT-IN に設定されていることを確認します (変換言語の選択を参照)。
2. 「**ファイル**」メニューから「**FlowForce Server にデプロイ**」を選択します。マッピングのデプロイダイアログボックスが開かれます。



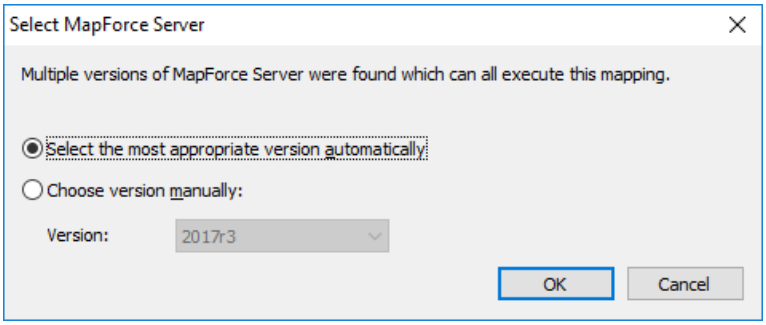
3. (下に記述されているとおり)デプロイの設定を入力し、「**OK**」をクリックします。**新しいジョブ作成のためにウェブブラウザを開く**チェックボックスが選択されている場合、FlowForce Server 管理インターフェイスが開かれ、すぐに FlowForce Server ジョブの作成を開始することができます。

以下のテーブルは、マッピングのデプロイダイアログボックスで使用することのできるマッピングデプロイ設定をリストしています。

設定	説明
サーバーとポート	FlowForce Server のサーバーホスト名 (または I.P. アドレス) とポートを入力します。これは FlowForce Server がデフォルトのポートと同じマシンで実行されている場合、**localhost** と **8082** である可能性があります。疑問がある場合は、FlowForce Server Web 管理インターフェイスにログインして、Web ブラウザーのアドレスバーに表示される I.P. アドレスとポートを確認してください。 接続エラーが生じた場合、FlowForce Server が実行されているマシンが入力接続を許可するように構成されていることを確認してください。
ユーザーとパスワード	ユーザー名とパスワードは、ログインドロップダウンリストの値 (次のオプション参照) により入力されます。ログインドロップダウンリストが **<Default>** または **Directly** に設定されている場合、FlowForce Server ユーザー名とパスワードを入力します。それ以外の場合、Windows ユーザー名とパスワードを入力します。ログインドロップダウンリストから Windows ドメイン名を選択します

ログイン	Windows Active Directory 統合が、FlowForce Server で有効化されている場合、このドロップダウンリストから、Windows ドメイン名を選択し、ユーザーとパスワードフィールドにWindows 資格情報を入力します（前のオプションを参照してください）。
パス	**「参照」**をクリックして、FlowForce Server コンテナーの階層内でマッピング関数が保存されるパスを選択します。デフォルトでは、パスはFlowForce Server の**/ public** コンテナーです。 ダイアログボックスから、デプロイ名の選択することができ、また、新規のコンテナーを作成することもできます。必要なFlowForce Server パーミッションと特権が与えられている場合、既存のコンテナーとマッピングを削除できることもできます。  *デプロイ名の選択ダイアログボックス*
デプロイ前にマッピングを保存する	保存されていないマッピングをデプロイする時にこのオプションを使用することができます。デプロイ前にマッピングを保存する場合このチェックボックスを選択します。
新しいジョブを作成するためにブラウザを開く	このチェックボックスを選択した場合、FlowForce Server Web 管理インターフェイスがデプロイ後開かれ、FlowForce Server ジョブの作成の開始をすることができます。

マッピングをデプロイするサーバーでFlowForce Server 管理下で、複数のバージョンのMapForce Server を作動している場合、（Windows サーバーのみに適用することができます）このマッピングを実行するMapForce Server のバージョンを指定するようにプロンプトされます。



メモ: FlowForce Server インストールディレクトリに tool ファイルが含まれている場合、FlowForce Server の管理下で実行される各 MapForce Server のバージョンのために、ダイアログボックスが表示されます。デフォルトでは、MapForce Server をFlowForce Server の一部としてインストールすると、 MapForce Server .tool ファイルは、このディレクトリに自動的に追加されます。FlowForce 内でファイルが保管されるパスは以下のとおりです :**C:\Program Files\Altova\FlowForceServer2017\tools**。FlowForce Server 管理下での他のMapForce Server バージョンを作動する場合、 tool ファイルを手動で上記のディレクトリにコピーする必要があります。MapForce Server の tool ファイルは以下に保管されています :**C:\Program Files \Altova\MapForceServer2017**

11.4 RaptorXML Server を使用して自動化する

RaptorXML Server (今後は、略してRaptorXML) は、Altova 第3世代の超高速なXML とXBRL のためのプロセッサーです。最新の標準とパラレル コンピューティング環境のために最適化されています。クロスプラットフォームに対応可能で、エンジンは今日のマルチコアコンピューティングを有効に利用し、XML とXBRL データの高速処理を提供します。

RaptorXML には、Altova ダウンロードからダウンロードしてインストールすることのできる複数のエディションがあります :

- RaptorXML Server は、XML、XML スキーマ、XSLT、XPath、XQuery、などへのサポートを搭載した高速なXML 処理エンジンです。このエディションは、FlowForce Server インストールパッケージの一部です。
- RaptorXML+XBRL Server は、XBRL の一連の標準を処理および検証する追加機能を使用して、RaptorXML Server の全ての機能をサポートします。

制約事項 :

- XML 署名はサポートされていません
- COM インターフェイスを介してグローバルリソースはサポートされていません
- ODBC とADO データベース接続はWindows のみでサポートされています。他のオペレーティングシステムでは、JDBC を使用する必要があります。

XSLT 1.0 または 2.0、またはin XQuery、でコード生成するには、MapForce は、生成の際に選択する出力フォルダーに保存される`DoTransform.bat` と呼ばれるバッチファイルを作成します。バッチファイルの実行は、RaptorXML Server を呼び出し、サーバー上でXSLT/XQuery 変換を実行します。

サーバー上の他の出力のために、MapForce マッピングを実行または自動化するには、Altova MapForce Server と FlowForce Server を参照してください。

メモ: 内蔵のエンジンを使用して、XSLT とXQuery コードをプレビューすることができます。

11.5 MapForce コマンドラインインターフェイス

一般的なコマンドライン構文 :

MapForce.exe *ファイル名* [*/target* [*outputdir*] *オプション*

- 角かっこ**[...]** 任意のパラメーターを表します。
- 中かっこ**{...}** 複数の選択肢を含むパラメータのグループを表します。
- **パイプ**シンボル**|** は、OR を表します。例 : /XSLT または/JAVA

コマンドラインの実行に成功した場合、MapForce.exe は終了コード 0 を返します。他の値は失敗を意味します。バッチファイル内のIF ERRORLEVEL コマンドを使用してこのコードをチェックすることができます。

ファイル名
ロードするMFD またはMFP ファイル。**パスまたは、ファイル名にスペースが含まれると、パス /ファイル名を引用符で囲んでください。すなわち、 "c:\ Program Files\ ...\ Filename"**

ターゲット
コード生成ターゲット :

/COMPILE :*compileoptions*	**MapForce Server 実行ファイル**(.mfx)にマッピングをコンパイルする *compileoptions* **JDBC**、**NOXMLSIGNATURES** **JDBC** は、データベースコンポーネント設定ダイアログボックス内で定義されたJDBC ドライバーとデータベースURL を使用するを変換します。t **NOXMLSIGNATURES** は、MapForce Server 実行ファイル内の(MapForce Server によりサポートされていない)デジタル署名 の生成を抑制します。
/GENERATE	プロジェクトファイル内の全てのマッピングのためにgenerates プロジェクトコードusing 現在のフォルダー設定 . プロジェクトファイル *MFP は、ファイル名として使用される必要があります。
/XSLT	XSLT 1.0 コードを生成します
/XSLT2	XSLT 2.0 コードを生成します
/XQuery	XQuery 1.0 コードを生成します
/JAVA	Java アプリケーションを生成します
/CS	マッピング設定の構成を使用してC# アプリケーションを生成します
/CS:*csoptions*	指定されたオプションを使用してC# アプリケーションを生成します *csoptions* { **VS2010** | **VS2008** | **VS2005** } **VS2010** は、Visual Studio 2010 のためのソリューションファイルつきのC# アプリケーションを生成します **VS2008** は、Visual Studio 2008 のためのソリューションファイルつきのC# アプリケーションを生成します

VS2005 は、Visual Studio 2005 *のためのソリューションファイルつきの C# アプリケーションを生成します*

/CPP は、マッピング設定の構成を使用して C++ アプリケーションを生成します

/CPP:*cppoptions* は、指定されたオプションを使用して、C++ アプリケーションを生成します

cppoptions { **VC10** | **VC9** | **VC8** },{ **MSXML** | **XERCES** | **XERCES3** }, { **LIB** | **DLL** },{ **MFC** | **NoMFC** }

VC10 は、Visual Studio 2010 *のためのソリューションファイルつきの C++ アプリケーションを生成します*

VC9 は、Visual Studio 2008 *のためのソリューションファイルつきの C++ アプリケーションを生成します*

VC8 は、Visual Studio 2005 *のためのソリューションファイルつきの C++ アプリケーションを生成します*

MSXML は、MSXML 6.0 を使用して C++ コードを生成します。

XERCES は、Apache Xerces 2.x (2.6 *または以降*) を使用して C++ コードを生成します。

XERCES3 は、Apache Xerces 3.x を使用して C++ コードを生成します。

LIB は、静的*ライブラリのために* C++ コードを生成します。

DLL は、動的*リンクライブラリのために* C++ コードを生成します。

MFC は、MFC (Microsoft Foundation Classes) サポートを使用して、C++ コードを生成します。

NoMFC は、MFC サポートを使用せずに、C++ コードを生成します。

outputdir
outputdir 内に生成されたマッピングコードが置かれるディレクトリは任意です。出力パスが与えられない場合、作業 *(現在)* のディレクトリが使用されます。

オプション
複数のオプションを指定します :

/GLOBALRESOURCEFILE *grfilename* 指定されたグローバルリソースファイル内で定義されたグローバルリソースを使用します

/GLOBALRESOURCECONFIG *grconfig* 指定されたグローバルリソースの構成を使用します

/LIBRARY *libname* コード生成ターゲット言語と共に使用して、追加関数ライブラリを指定します。このオプションは複数のライブラリをロードするために、1度以上指定することができます。これらのライブラリは一時的に (1度限りの実行の際) ライブラリにツール -> オプション -> ライブラリから追加されます。

/LOG *logfilename* ログファイルを生成します。*logfilename* はフルパス名であることができます。すなわち、ログファイルのディレクトリとファイル名です。しかし、フルパスが与えられる場合、ディレクトリはログファイルを生成するために存在しなければなりません。

ファイル名を指定すると、*(ファイルは outputdir* ディレクトリに置かれます。

メモ:
コマンドラインパラメーターの1つを相対パスで入力することは、作業ディレクトリに対して相対的とみなされます。作業ディレクトリはMapForce を呼び出しているアプリケーションの現在のディレクトリをさします。これは、MFD ファイルfilename、outputdir、logfilename、およびglobalresourcefilename のパスに適用されます。

絶対パスの定義は作業ディレクトリの無視を引き起こします。絶対パスは与えられたとおり使用されます。

コマンドラインでは、バックスラッシュや引用符 (\") を末尾で使用することを可能な限り避けてください。例 : "C:\My directory\\". これらの文字は、コマンドラインパーサーにより、テキストの二重引用符として解釈されます。コマンドライン内でスペースが発生すると、バックスラッシュ\\ を使用し、引用符も使用します ("c:\My Directory\\")。またはスペースの使用を避け、引用の使用を避けることもできます。例 :c:\MyDirectory。

例 :

MapForce.exe *filename* は、MapForce **を開始し、***filename* により定義されたファイルを開きます。

I)
すべてのXSLT ファイルを生成し、ログファイルを出力します。
MapForce.exe *filename.mfd* **/XSLT** *outputdir* **/LOG** *logfilename*

II)
Java アプリケーションを生成し、ログファイルを出力します。
MapForce.exe *filename.mfd* **/JAVA** *outputdir* **/LOG** *logfilename*

III)
C# アプリケーションを生成し、ログファイルを出力します。
MapForce.exe *filename.mfd* **/CS** *outputdir* **/LOG** *logfilename*

IV)
アプリケーションオプション内で定義されたコード生成設定を使用し、C++ アプリケーションを生成し、ログファイルを出力します。
MapForce.exe *filename.mfd* **/CPP** *outputdir* **/LOG** *logfilename*

V)
/CPP スイッチを使用して、C++ コンパイラーオプションをオーバーライドし、C++ アプリケーションを生成します。
MapForce.exe *filename.mfd* **/CPP:(MSXML|XERCES),(LIB|DLL),(MFC|NoMFC)** *outputdir* **[/LOG** *logfilename* **]**

MapForce.exe *filename.mfd* **/CPP:MSXML,LIB,MFC**
最初の選択すべてを使用して、C++ アプリケーションを生成する場合。このサンプルでは :

- C++ のためにコンパイル
- MSXML を使用
- 静的ライブラリのためにコードを生成
- MFC のサポートのためのコード生成

MapForce.exe *filename.mfd* **/CPP:XERCES,DLL,NoMFC** *outputdir* **/LOG** *logfilename*
2 番目の選択すべてを使用して、C++ アプリケーションを生成する場合。このサンプルでは :

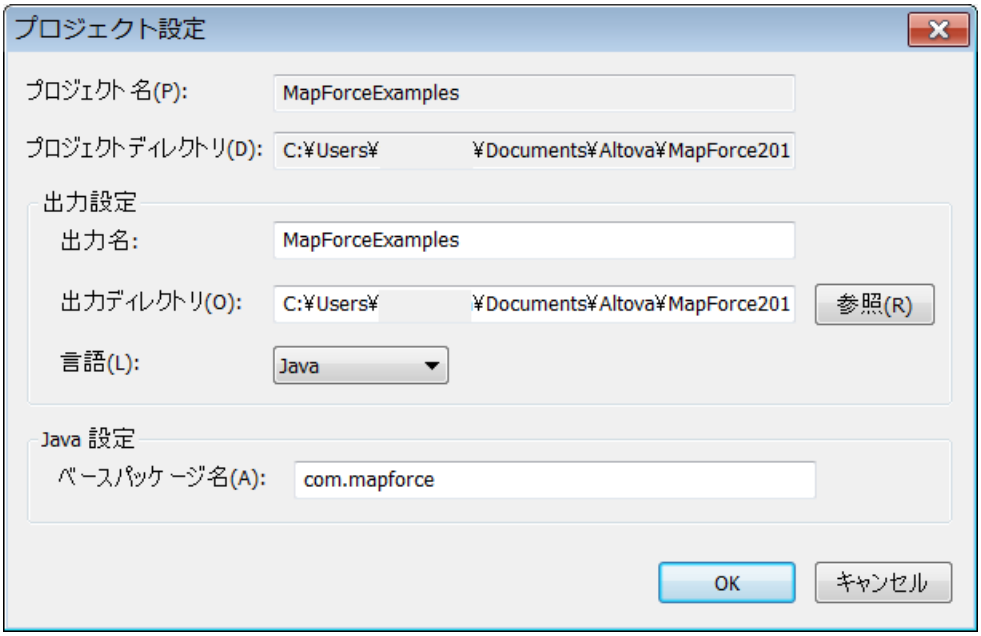
- C++ のためにコンパイル
- XERCES の使用
- 動的ライブラリのためにコードを生成
- MFC をサポートしないためのコード生成
- ログファイル名を持つ出力ディレクトリ内にログファイルを作成

VI)
指定された構成のためのグローバルリソースファイルのグローバルリソースを使用してすべての出力ファイルを生成します。
Mapforce.exe *filename.mfd outputdir* **/ GLOBALRESOURCEFILE** *globalresourcefilename* **/ GLOBALRESOURCECONFIG** *configurationname*

VII)
現在のプロジェクトフォルダー設定を使用して、**プロジェクトファイル**内の全てのマッピングのための、プロジェクトコードアプリケーションを生成します。
MapForce.exe *filename.mfp* **/GENERATE /LOG** *logfilename*

プロジェクト全体のためにコードを生成する場合、プロジェクトファイル名、例 :**MapForceExamples.mfp**、は *filename* として使用される必要があります。
各フォルダ内のコード生成言語およびマッピングのための出力パスは、*outputdir* が指定**されていない**場合、**各**フォルダーのフォルダープロパティダイアログボックスにより提供されます。

プロジェクトコードを生成するために、GENERATE を使用して、*outputdir* パラメーターが使用される場合、**プロジェクトルート**フォルダーの *Outputdir* ディレクトリのパスは、コマンドライン内に与えられるエントリにより上書きされます。新しいフォルダのプロパティダイアログボックス内の "デフォルトプロジェクト設定を使用 " ラジオボタンを選択する場合に使用されるディレクトリです。



VIII)
プロジェクトファイル内の全てのマッピングのために、Java でプロジェクトコードを生成する場合。
MapForce.exe *filename.mfp* **/JAVA /LOG** *logfilename*

フォルダープロパティ設定により定義されたコード生成言語は無視され、全てのマッピングでは Java が使用されます。

IX)
実行可能な JAR ファイルを用いて、既にコンパイルされた (Java) マッピング プロジェクトの出力ファイルを生成する場合。コマンドライン内で入力および出力ファイルを定義する場合。
java -jar *mappingfile.jar* /**InputFileName** *inputfilename* /**OutputFileName** *outputfilename*

メモ: /InputFileName と /OutputFileName **パラメーター** は、コマンドラインの実行内のパラメーターの使用を許可する MapForce マッピング内の特別な入力コンポーネントの名前です (次を参照してください: マッピングにパラメーターを与える)。

Chapter 12

Map のカスタム化

12 Map のカスタム化

このセクションではAltova グローバルリソースとの作業、、マッピングの出力、マッピングドキュメントのカスタム化、およびカタログファイルとの作業方法について説明します。

12.1 MapForce オプションの変更

MapForce 内の一般および他の優先順位を変更する方法は、以下の通りです：

- **「ツール」**メニューから **「オプション」**をクリックします。

使用することのできるオプションは、以下に表示されているとおりです。

ライブラリ

- MapForce にカスタム関数 ライブラリを追加または削除するには以下を参照してください（カスタム XSLT 1.0 または 2.0 関数のインポート、カスタム XQuery 1.0 関数のインポート、カスタム Java と .NET ライブラリのインポート）。

全般

- MapForce の開始時にエコを表示する、または、非表示にするかを指定します。
- マッピング ペイン内の背景のグラデーションでの表示を有効化または無効化します。
- **注釈の表示 ...**は、注釈をサポートするコンポーネントに適用されます（例えば、XML スキーマ、EDI）。注釈のテキストが複数の行を含む場合、このオプションを有効化することにより、コンポーネントの最初の N 行のみが表示されます。N は指定される値です。この設定はコンポーネント内で表示される SELECT ステートメントにも適用されます。
- 新しいコンポーネントのためのデフォルトの文字エンコードを定義します。この設定は各コンポーネントのために個別に変更されることもできます（コンポーネントの設定の変更 を参照）。
- 出力ペイン内でマッピングの結果をプレビューする際の実行のタイムアウトを定義します。
- 出力ペイン内でマッピングの結果をプレビューする際、（デフォルトのオプションである）マッピングの出力が一時的ファイルに書き込まれるか、または、最終ファイルに直接書き込まれるかを指定します。

  警告: "最終出力ファイルに直接書き込む" オプションが有効化されている場合、確認を必要とすることなく、出力ファイルは上書きされます。

- 大きな XML とテキストファイルを生成するマッピングをプレビューする場合、出力ペイン内のテキストの表示の最大値を指定します（出力のプレビュー を参照）。

編集

- マウスでドラッグする際にコンポーネントまたは関数が他のコンポーネントと共に整列するかを指定します（コンポーネントの整列 を参照）。
- 有効化されると、**スマートコンポーネントの削除** オプションは、削除された接続を "記憶" します（コンポーネントの削除後、接続を保持する を参照）。

メッセージ

- "このメッセージを表示しない" オプションにより以前に無効化されたメッセージの通知を再有効化します。

生成

- プログラムのコード生成中に適用できる IDE とコンパイラーの設定を定義することが有効化できます。（コードジェネレーターのオプション を参照）。

XBRL

- XBRL コンポーネントの特定の表示設定を定義します（一般 XBRL オプションを参照）。

デバッガー

- マッピングのデバッグに適用できる設定を定義します（デバッガーの設定 を参照）。

データベース

- **DB Query** タブ内のデータベースをクエリするとき適用することのできる設定を定義することができます（次を参照してください：データベースの参照とクエリ）。

12.2 Altova グローバルリソース

グローバルリソースはAltova 製品間において生産性を向上させるための機能で、現在以下のAltova 製品にて提供されています：:XMLSpy、MapForce、StyleVision、DatabaseSpy。Altova MissionKit を使用中のユーザーも、該当する製品でこの機能へアクセスすることができます。

一般的な利用方法：

- データ処理を行う様々なAltova アプリケーションを使用するワークフローの定義
- あるアプリケーションから別のアプリケーションを呼び出し、データ処理を行ったあと、元のアプリケーションへデータを戻すといった作業
- 入出力データ ファイルやデータベース の場所をグローバルリソースとして定義
- ランタイムにおいてグローバルリソースを切り替えることで、開発リソースとデプロイリソースを切り替え
- 品質管理の試験における「もしも」シナリオ

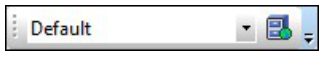
グローバルリソースの定義ファイルはGlobalResources.xml で、C:\Documents and Settings\UserName\My Documents\Altova\ 以下に収められています。これはグローバルリソースを使用する全てのAltova アプリケーションで共通したデフォルトの場所となります。そのため、グローバルリソースに対して行われた変更は、自動的に他のアプリケーションでも利用可能になります。ファイル名や位置を変更することもできます。詳細については グローバルリソース - プロパティ を参照ください。

一般的なメカニズム：

- グローバルリソースはアプリケーション内部にて **定義され**、自動的に保存されます。
- グローバルリソースはコンポーネントに対して **割り当てられ**、内部のデータを動的に扱うことができます。
- グローバルリソースはアプリケーション内部にて **呼び出され**、ランタイムにてリソースの切り替えを行うことができます。

このセクションでは…\MapForceExamples\Tutorial\ フォルダーに収められている既存のマッピングにてグローバルリソースを定義ならびに使用する方法を説明します。

グローバルリソースツールバーを利用可能にする：
メニューオプションの「ツール| カスタマイズ」を選択し、ツールバータブをクリックした後、グローバルリソースの管理のチェックボックスにチェックを入れます。この操作によりグローバルリソースツールバーが表示されます。



コンボボックスによりグローバルリソースの切り替えを行うことができます。"Default" エントリーは常に選択することができます。グローバルリソースアイコン をクリック (または「ツール| グローバルリソース」を選択) することで、グローバルリソースダイアログボックスが表示されます。

12.2.1 グローバルリソース - ファイル

MapForce のグローバルリソースでは：

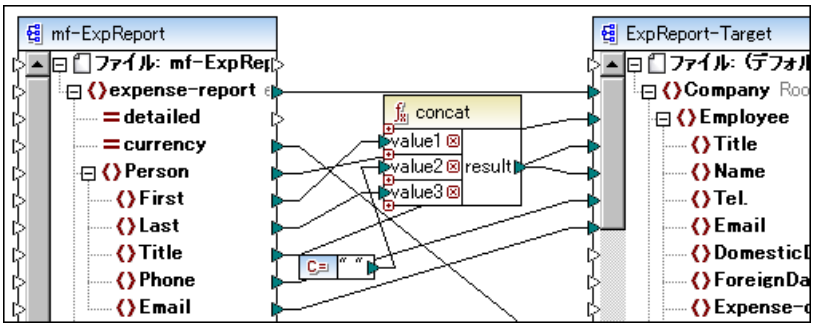
- XML、XML スキーマ、Text/CSV、データベース、EDI、Excel 2007 以降の ファイルといった、任意の入力 / 出力コンポーネントファイルをグローバルリソースとして定義することができます。

このセクションの目的：

- ソースコンポーネント入力ファイルmf-ExpReport をグローバルリソースとして定義する
- ランタイムにて **入力データ**となる 2 つのXML ファイルを **切り替え**、出力プレビュータブに表示されるXML 出力を確認する。

  このセクションでは…\MapForceExamples\Tutorial\ フォルダーに収められているTut-ExpReport.mfd ファ

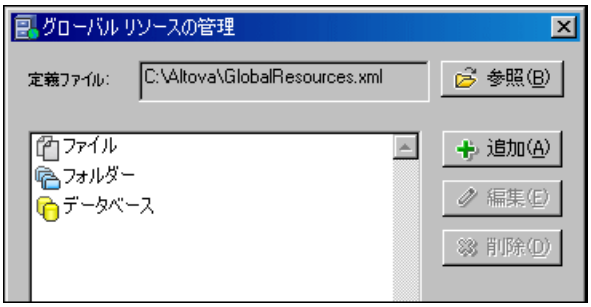
イルを使用します。



12.2.1.1 グローバルリソースの定義 /追加

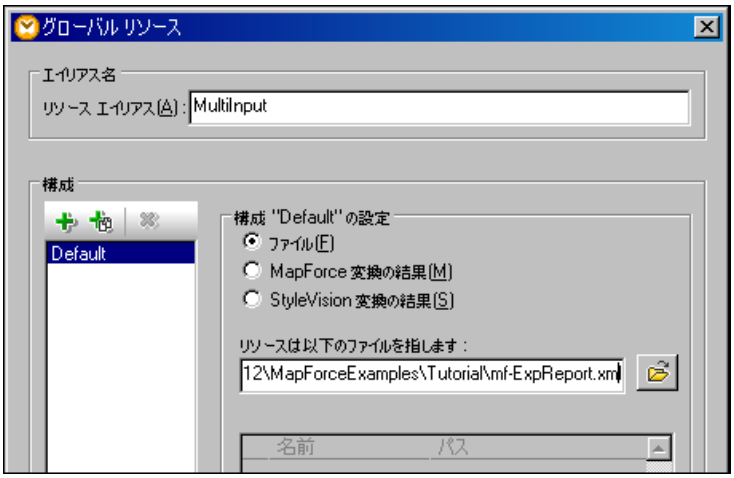
グローバルリソースファイルの定義 / 追加：

1. グローバルリソースアイコン をクリックして、ダイアログボックスを開きます。

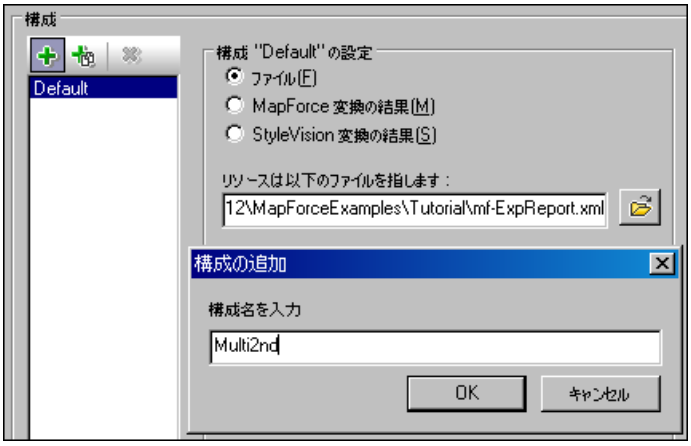


これがグローバルリソースが空の状態です。

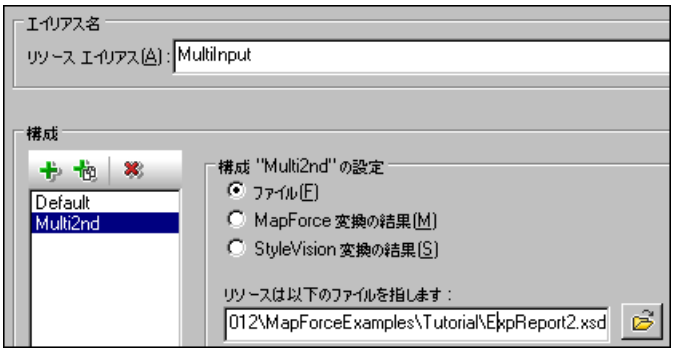
2. **「追加」**ボタンをクリックして、ポップアップから**ファイル**を選択します。
3. リソースエイリアスの名前（例：**MultiInput**）を入力します。
4. フォルダーを開くアイコンをクリックし、Default（デフォルト）の入力ファイルとなるXML ファイル（例：**mf-ExpReport.xml**）を選択します。



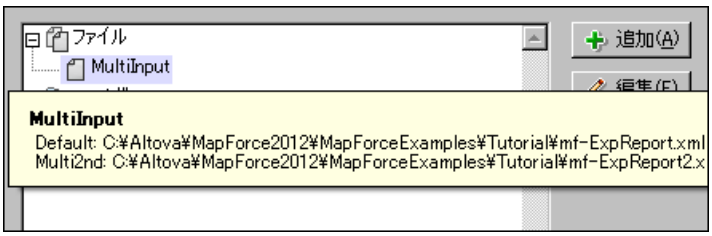
5. **構成**グループにある追加ボタン をクリックして、現在のエイリアスに対して新たな構成を追加します。構成のコピーアイコン を使用することで選択された構成をコピーして、新しい名前で保存することもできます。



6. 構成の名前（例えば**Multi2nd**）を入力し、「OK」をクリックすることで確定します。
Multi2nd が構成リストに追加されます。
7. フォルダーを開くアイコンを再びクリックし、Multi2nd 構成にて入力ファイルとなるXML ファイル（例：**mf-ExpReport2.xml**）を選択します。



8. **「OK」**をクリックしてリソースの定義を完了します。
MultiInput エイリアスがグローバルリソースのファイルセクションに追加されました。
マウスカーソルをエイリアスエントリーへマウスオーバーすることで、その定義がツールチップにて表示されます。



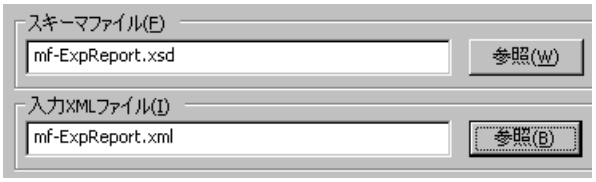
9. **「OK」**をクリックして確定します。
この操作によりグローバルリソースの定義が完了します。次のステップでは、グローバルリソースの割り当てについて説明します。

12.2.1.2 グローバル リソースの割り当て

グローバルリソースをコンポーネントへ割り当てる
グローバルリソースが定義できたので、グローバルリソースをコンポーネントへ割り当てます。つまり、mf-ExpReport.xml ファイルがマッピングのソースファイルとして使用されます。

1. **mf-ExpReport** コンポーネントをダブルクリックして入力 XML ファイルフィールドの隣にある参照ボタンをクリックし

ます。

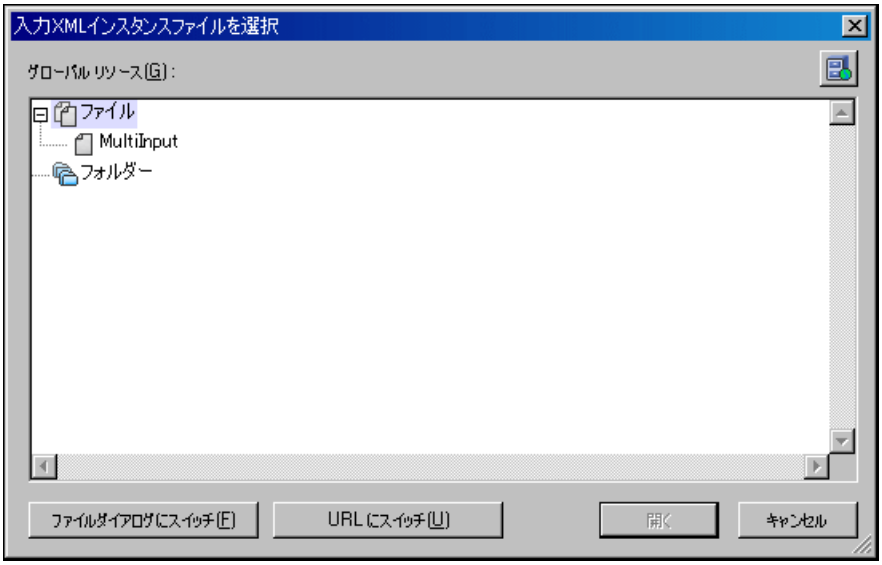


この操作により、「入力 XML インスタンスファイルを選択」ダイアログボックスが表示されます。

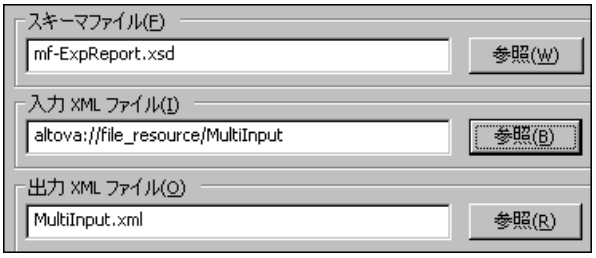
2. ダイアログボックスの下部にある**グローバルリソースにスイッチ**ボタンをクリックします。



3. 割り当てを行うリソース（この場合は **MultiInput** ）を選択して、開くをクリックします。



メモ：コンポーネントの入力 XML ファイルフィールドに、リソースに対する参照 **altova://file_resource/MultiInput** が表示されます

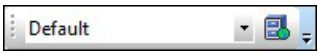


4. 「**OK**」をクリックして、コンポーネントに対するリソースの割り当てを完了します。

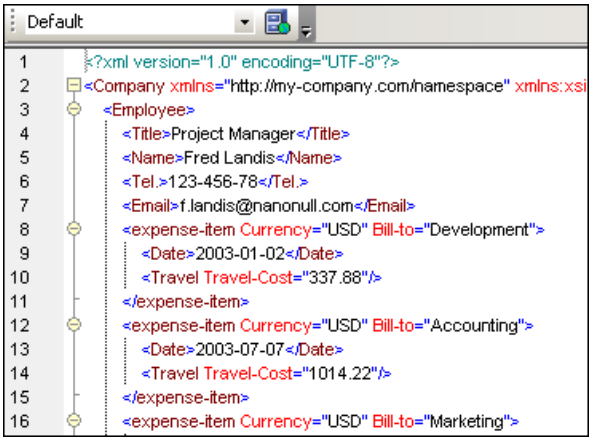
次のステップでは、グローバルリソースの使用 /アクティベートを行います。

12.2.1.3 グローバル リソースの使用 / アクティベート

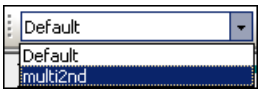
グローバルリソースの使用 / アクティベート
これまでの操作により **Default** 構成に対して定義された**MultiInput** エイリアスがアクティブになっています。これはグローバルリソースアイコンバーの表示が**"Default"** となっていることで確認することができます。



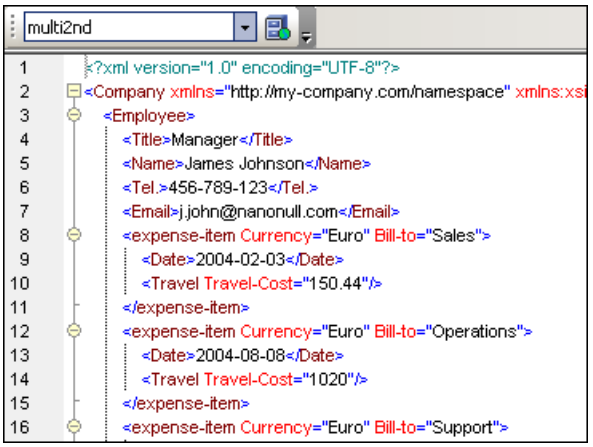
1. 出力タブをクリックして、マッピングの結果を確認します。



2. **マッピング**タブをクリックして、マッピングビューへ戻ります。
3. グローバルリソースコンボボックスをクリックして、コンボボックスから**multi2nd** を選択します。



4. 出力タブをクリックして、新たな結果を確認します。
グローバルリソースコンボボックスをクリックして、コンボボックスからmulti2nd を選択します。



メモ:
現在アクティブになっているグローバルリソース (グローバルリソースツールバーにある**Multi2nd**) により、マッピング結果が決定されます。コードの生成を行った場合でも同様のことが起こります。

12.2.2 グローバルリソース - フォルダー

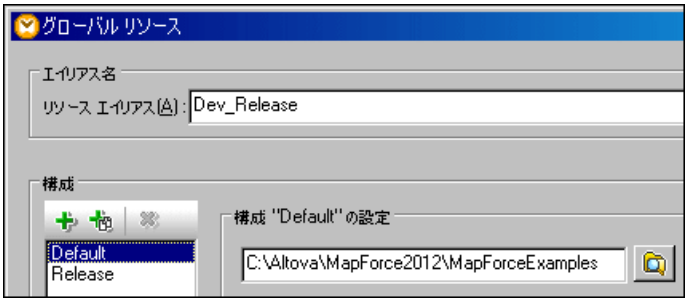
フォルダーをグローバルリソースとして定義することも可能で、例えば開発用とリリース用といった異なるフォルダーに含まれているファイルを参照するよう入力コンポーネントを定義することができます。

MapForce ではXSLT やその他のプログラムコードを生成する場合、ターゲットとなるフォルダーを指定するよう毎回促されるため、出力コンポーネントに対するフォルダーの定義はそこまで有効ではありません。

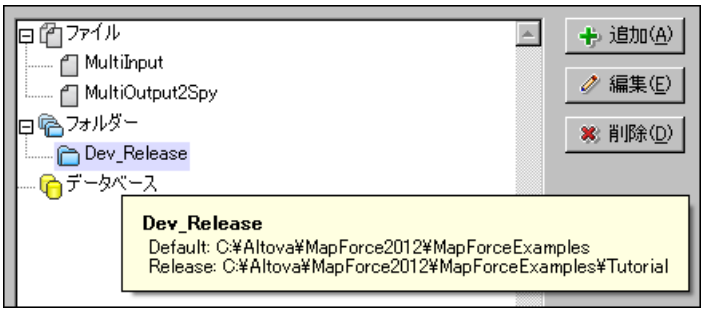
このセクションでは…\MapForceExamples\Tutorial\ フォルダーに収められているglobal-**フォルダー。mfd** ファイルを使用します

グローバルリソースフォルダーの定義 /追加

1. グローバルリソースアイコン をクリックして、ダイアログボックスを開きます。
2. **「追加」** ボタンをクリックして、ポップアップから**フォルダー**を選択します。
3. リソースエイリアスの名前 例：**Dev_Release**）を入力します。
4. フォルダーを開くアイコンをクリックし、"Default" の入力フォルダーとなる .\MapForceExamples を選択します。



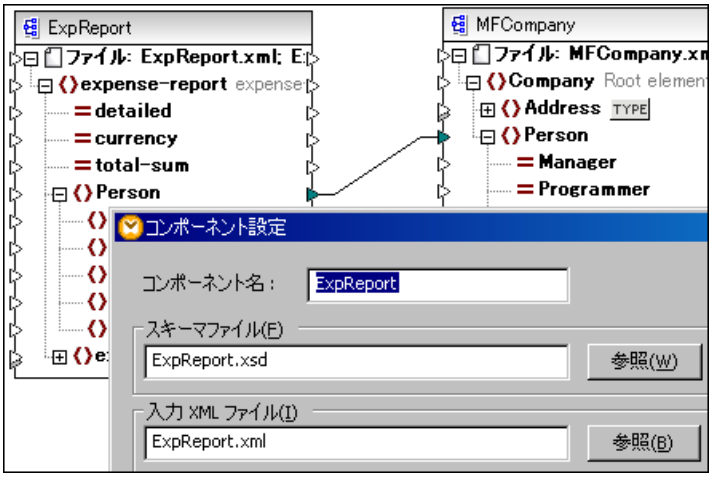
5. **構成**グループの追加ボタン をクリックして、現在のエイリアスに対して新たな構成の名前を入力 例：Release）して追加します。構成のコピーアイコンglob-copyic を使用することで選択された構成をコピーして、新しい名前にて保存することができます。
6. フォルダーを開くアイコンをクリックし、"Release" の入力フォルダーとなる .\MapForceExamples\Tutorial を選択します。



7. **「OK」** をクリックして、グローバルフォルダーの定義を完了します。

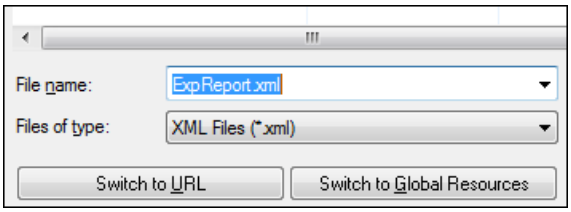
グローバルリソースフォルダーの割り当て

1. **ExpReport** コンポーネントをダブルクリックして、**入力 XML ファイルフィールド**の隣にある参照ボタンをクリックします

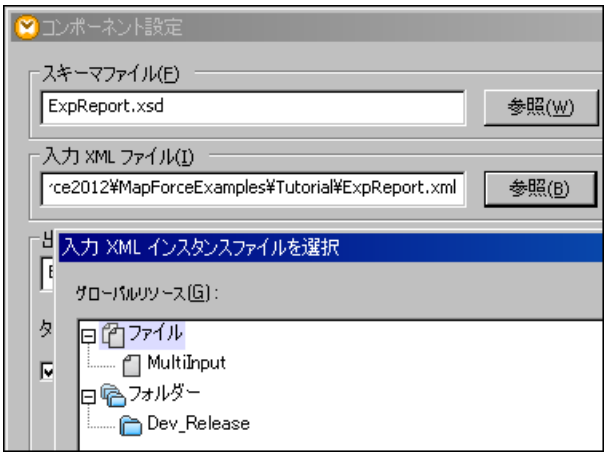


この操作により、XML インスタンスファイルを選択ダイアログボックスが表示されます。

2. ダイアログボックス下部にある**グローバルリソースにスイッチ**ボタンをクリックします。

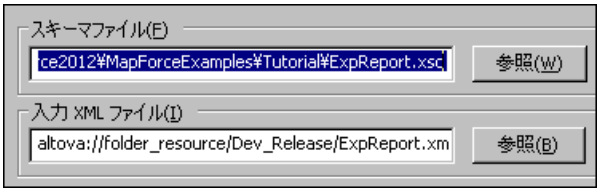


3. 割り当てを行うリソース（この場合では**Dev_Release**）をクリックして、**「OK」**をクリックします。



「ファイルを開く」ダイアログボックスが表示されます。

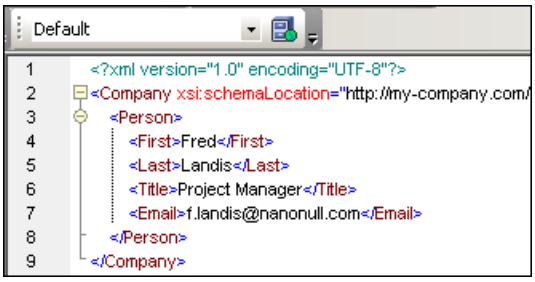
4. 両方のフォルダーにある、開発および本番リソース用の**リソース**ファイルとして使用することができるファイル（例えば**ExpReport.xml**）を選択して、**「OK」**をクリックすることでリソースフォルダーの割り当てを完了します。



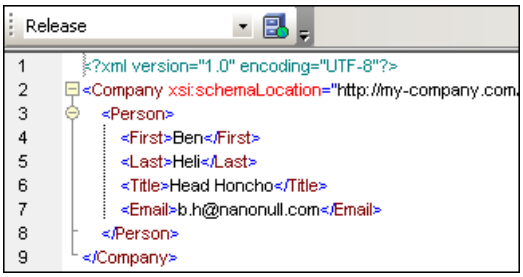
内容は違うものの、名前は同じファイルが両方のフォルダーに収められていることを確認してください。

ランタイムにてリソースフォルダーを変更する：

1. 出力タブをクリックして、変換の結果を確認します。
   Default 構成のフォルダー ...\MapforceExamples によりこの出力が作成されていることに注目してください。



2. マッピングタブをクリックしてマッピングウィンドウへ戻ります。
3. グローバルリソースコンボボックスをクリックして、"Relesae" エントリーを選択します。
4. 出力ボタンをクリックして、リリースグローバルリソースを使用し結果を確認します。

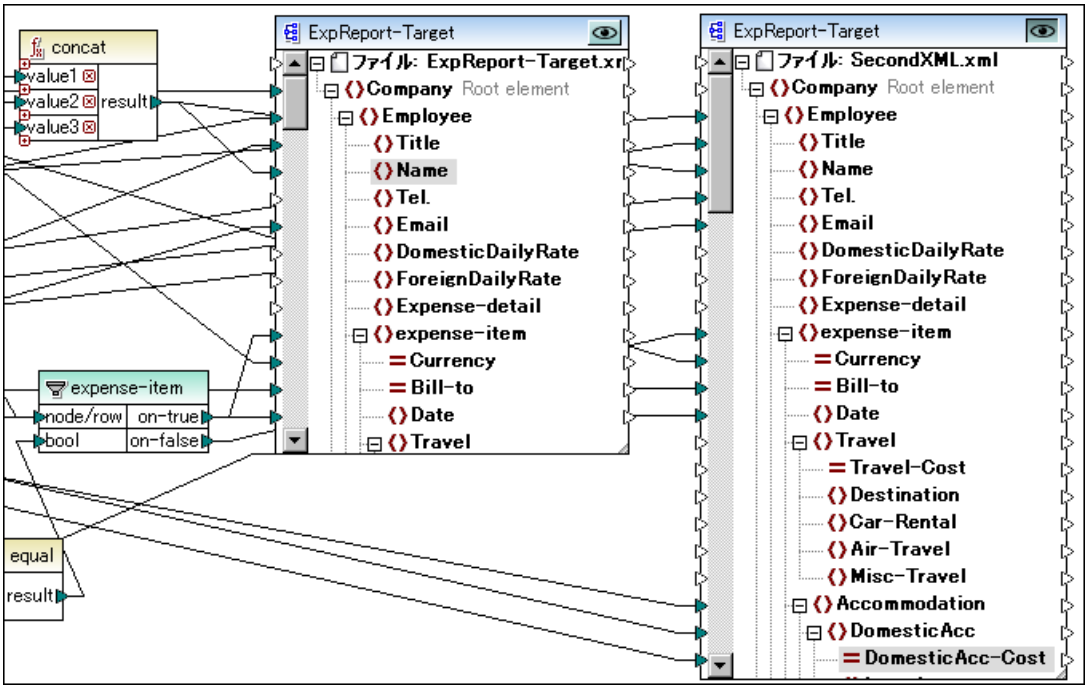


Release 構成のフォルダー ...\MapforceExamples/Tutorial のデータによる出力が表示されます。

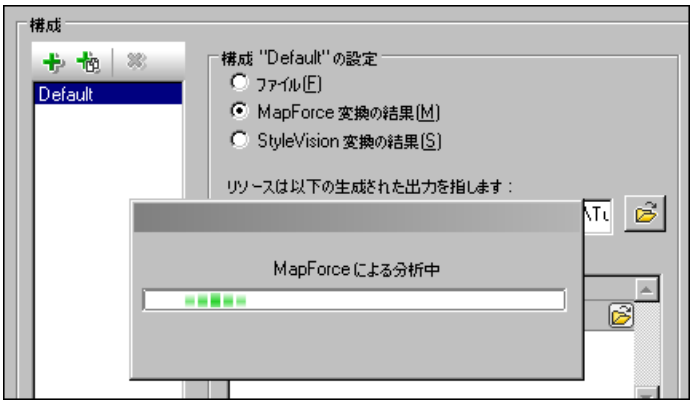
12.2.3 グローバルリソース - アプリケーションワークフロー

このセクションでは、2つのAltova アプリケーション間におけるワークフローを作成します。ワークフローではXMLSpy から MapForce が起動され、生成されたXML ファイル出力をXMLSpy へ戻すことで、更なる処理を行います。

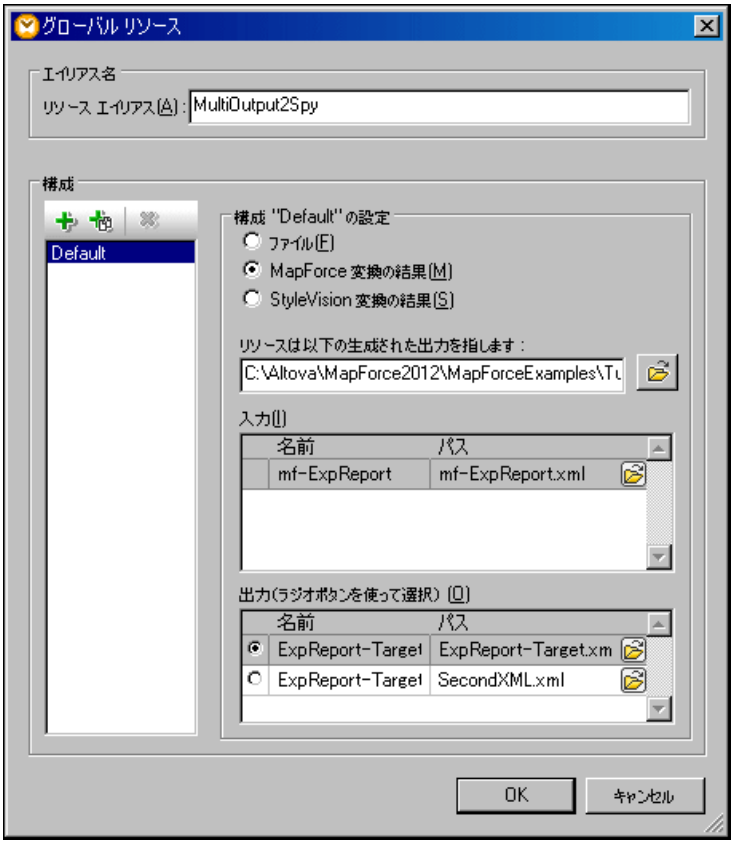
このマッピングでは 2つの出力コンポーネントが使用され、経費報告入力ファイルの旅費に関するファイルと、それ以外の費用に関するファイルと言うかたちでフィルタリングされた、2つの出力が作成されます。このセクションでは...\MapForceExamples\Tutorial\ フォルダーに収められているTut-**ExpReport-multi.mfd** マッピングファイルが使用されます。



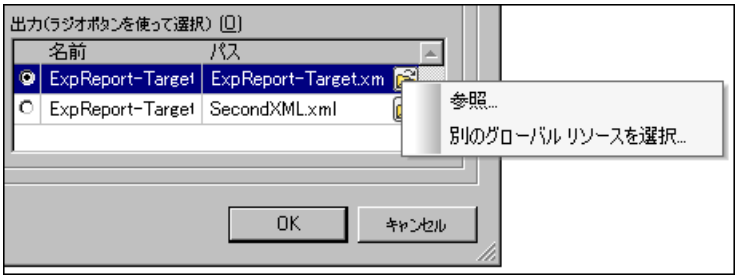
1. グローバルリソースアイコン をクリックして、ダイアログボックスを表示します。
2. **追加**ボタンをクリックして、ポップアップから**ファイル**を選択します。
3. リソースエイリアスの名前 例えば**MultiOutput2Spy**）**を入力します。**
4. **「MapForce 変換の結果」**ラジオボタンをクリックした後、ファイルを開くアイコンをクリックします。
5. **Tut-ExpReport-multi.mfd** マッピングを選択します。



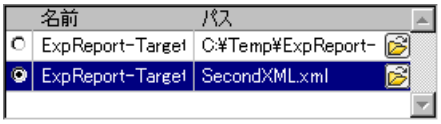
MapForce によりマッピングの解析が行われ、入力ならびに出力ファイルが別々のリストボックスにて表示されます。



6. **出力**セクションにある一番上のラジオボタンを選択します（選択されていない場合）。
出力ファイル名は**ExpReport-Target.xml** で、現在 **Default** の構成を定義しているということに注意してください。
7. アイコンをクリックして、出力ファイルの新しい場所を定義します。ポップアップメニューから参照 ...を選択します（例：C:\Temp）。

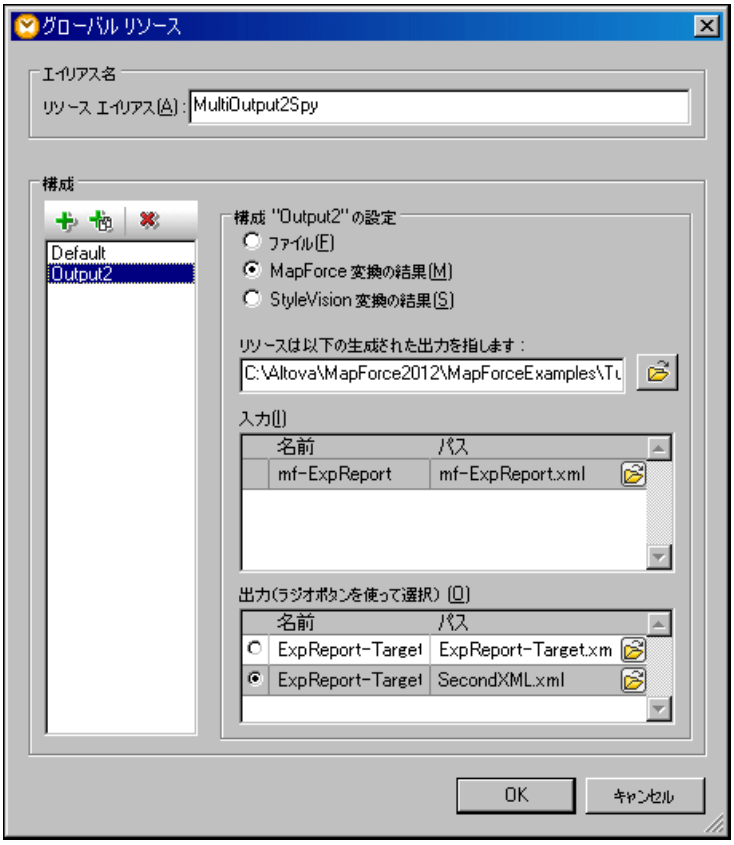


8. 新たな出力場所（例えばC:\Temp）を入力して、保存ボタンをクリックします。コンポーネント設定とは異なる場所も指定することができます。
9. ダイアログボックスの構成グループにある追加ボタン をクリックして、リソースエイリアスに新たな構成を追加します。
10. 構成の名前（例えば**Output2**）を入力し、ファイルを開くアイコンをクリックした後、**Tut-ExpReport-multi.mfd** を選択します。
11. 出力リストボックスにある2番目のラジオボタンを選択します。出力ファイル名が**SecondXML.xml** となっていることに注目してください。



12. 名前を付けて保存アイコン をクリックして、出力ファイルの新たな場所（例えば C:\Temp）をポップアップメニューから定義します。

メモ：ポップアップにて **「別のグローバルリソースを参照」** を選択すると、MapForce 出力をグローバルリソースとして保存することができます。つまり、マッピングの出力として作成されるファイルに対して、グローバルリソースの参照が行われます。



13. **「OK」** をクリックしてグローバルリソースを保存します。

新しいリソースエイリアスの **MultiOutput2Spy** がグローバルリソースの定義ファイルへ追加されます。

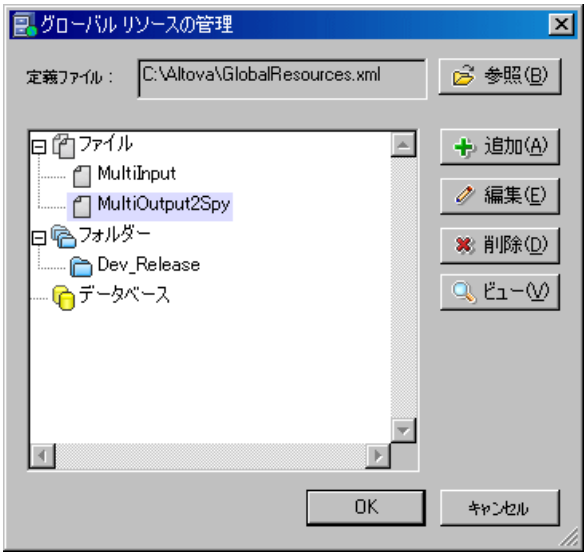


14. **「OK」** をクリックして定義を完了します。

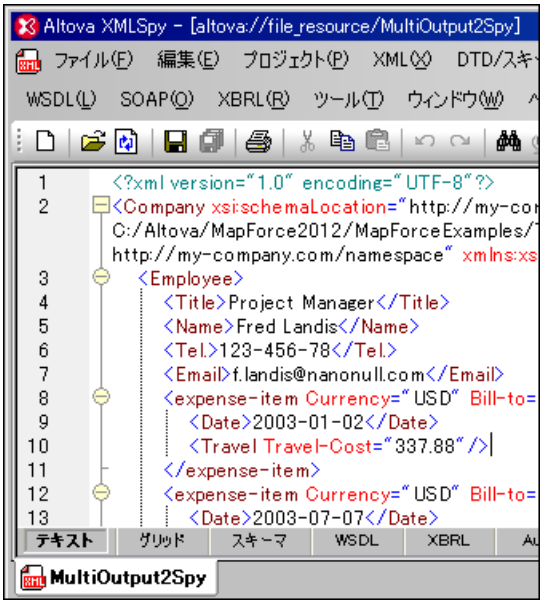
12.2.3.1 アプリケーションワークフローの起動

このセクションではXMLSpy におけるグローバルリソースの使用方法と MapForce の変換結果が返ってくるまでを説明します。

1. Start XMLSpy を起動して、MapForce が動作している場合は、(これらのアプリケーションがどのように動作するのか確認するため) MapForce を終了します。
2. XMLSpy にてメニューオプションの **「ツール | グローバルリソース」** を選択します。
3. **MultiOutput2Spy** エントリーを選択してビューボタンをクリックします



MapForce が変換を行なっているというメッセージボックスが表示され、変換結果がテキストビューウィンドウにて表示されます。



メモ:

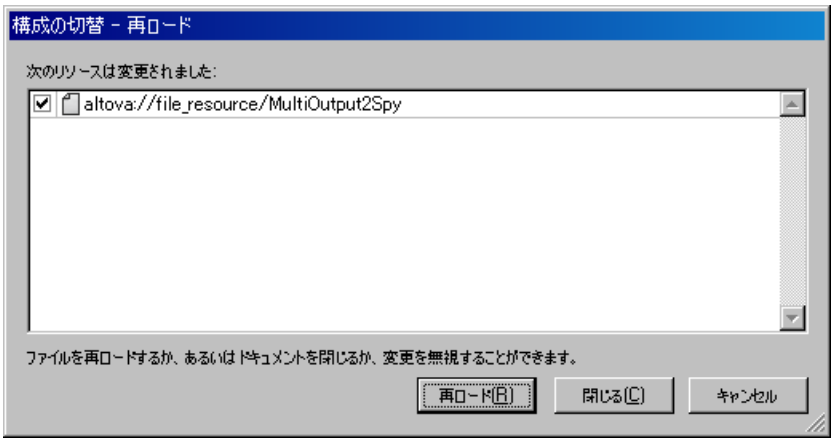
- 現在選択されている構成は **"Default"** です。
- リソースエイリアスの名前が、アプリケーションタイトルバーに表示されます：**altova://file_resource/**

MultiOutput2Spy

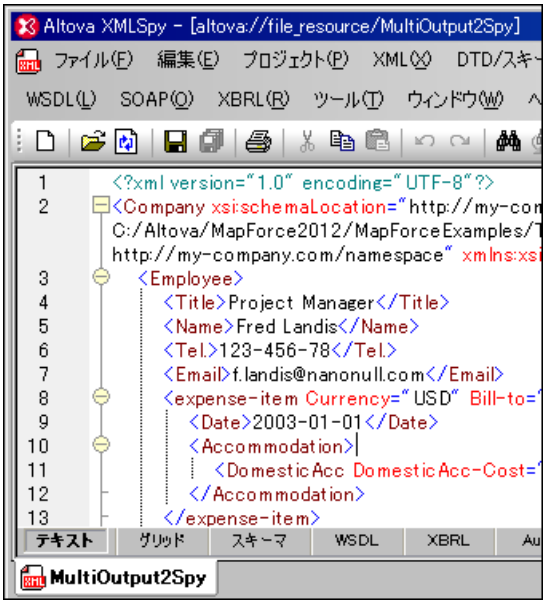
- 出力ファイルが MultiOutput2Spy という名前で開かれ、更に処理や編集などを行うことができます。
- **ExpReport-Target.xml** ファイルが C:\Temp フォルダーへコピーされます。

旅費以外の経費出力を取得する:

1. メニューオプションの「ツール| アクティブな構成」から "**Output2**" を選択します。
   以下のようなメッセージボックスが表示されます。



2. **再ロード**ボタンをクリックして、リソースにより定義されている 2 番目の出力ファイルを取得します。



変換結果がテキストビューウィンドウに表示され、それまで表示されていた "MultiOutput2Spy.xml" ファイルが上書きされます。

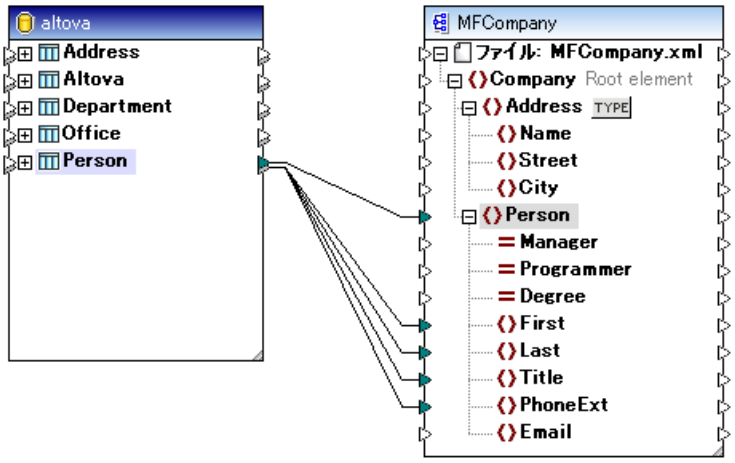
メモ:

- 現在選択されている構成は "**Output2**" です。
- 出力ファイルが MultiOutput2Spy という名前で開かれ、更に処理や編集などを行うことができます。
- **SecondXML.xml** ファイルが C:\Temp フォルダーへコピーされます。

12.2.4 グローバルリソース - データベース

データベースコンポーネントをグローバルリソースとして定義することで、例えば開発用とリリース用の異なるデータベースを参照することができます。データベースリソースはソースならびにターゲットコンポーネントとして使用することができます。

...\MapForceExamples\Tutorial\ フォルダーに収められている **"PersonDB.mfd"** ファイルを使用します。

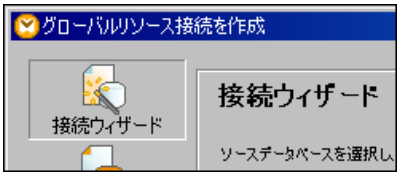


データベースグローバルリソースの定義 / 追加

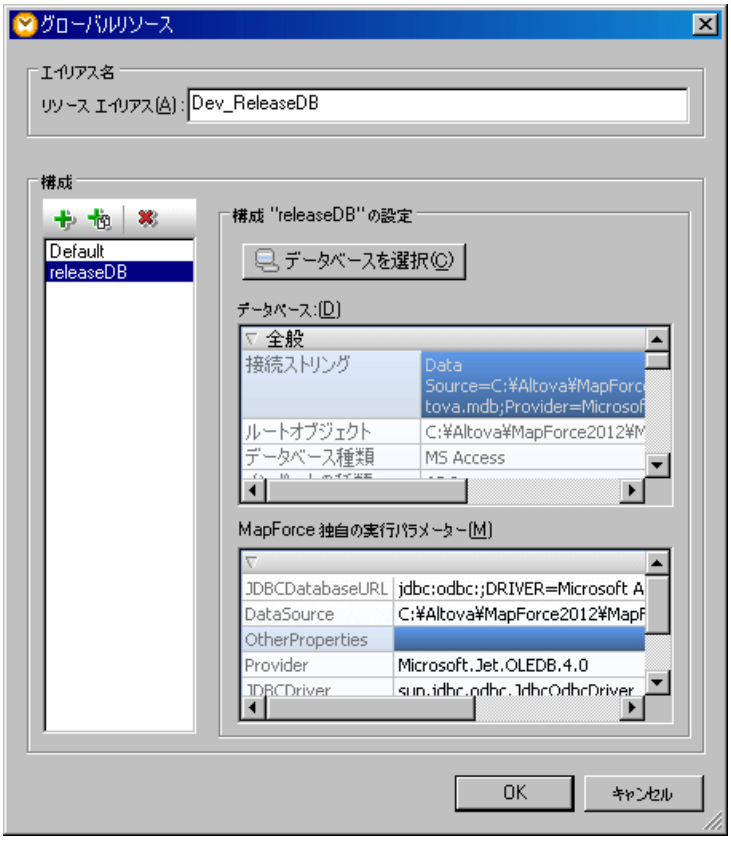
1. グローバルリソースアイコン をクリックして、ダイアログボックスを表示します。
2. **追加**ボタンをクリックして、ポップアップから**データベース**を選択します。
3. リソースエイリアスの名前 例えば **Dev_ReleaseDB**）を入力します。



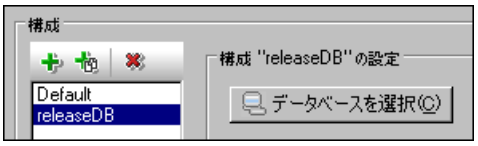
4. データベースを選択ボタンをクリックして、 接続ウィザードダイアログボックスにて 接続ウィザードボタンをクリックします。**\MapForceExamples** フォルダーにある**altova.mdb** ファイルを選択します。



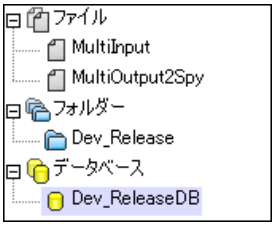
これが**開発用**のデータベースとなります。



5. 構成グループの**追加**ボタン をクリックして、現在のエイリアスに対して新たな名前（例えば **releaseDB**）の構成を追加します。



6. 再度データベースを選択ボタンをクリックして、リソースデータベース（例えば **..\MapForceExamples\Tutorial** フォルダーにある**altova.mdb**）を選択します。**「OK」**をクリックして、データベースリソースの定義を完了します。



メモ：

データベースを選択すると、**データベース**ならびに**MapForce 独自の実行パラメーター**リストボックスの内容を編集することができるようになります。例えば、パスワード フィールドをダブルクリックすることで、選択されているデータベースへ送信するパスワードを変更することができます。

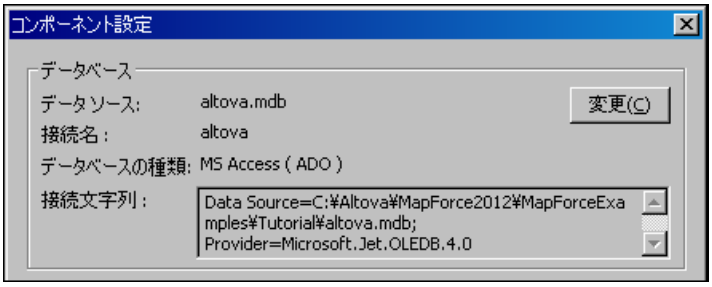
MapForce の実行パラメーターリストの内容には、データベースが選択された際にデフォルトの値が入力されます。これらのパラメーターはビルトイン実行エンジンによる**コード**の生成ならびに**出力プレビュー**の生成時に使用されます。

例えばIBM DB2 のような接続しているデータベースの種類により、「ルートオブジェクトの選択」ダイアログボックスが表示されることがあります。このダイアログでは、ルートオブジェクトを設定ボタンにより、ルートオブジェクトを直ちに

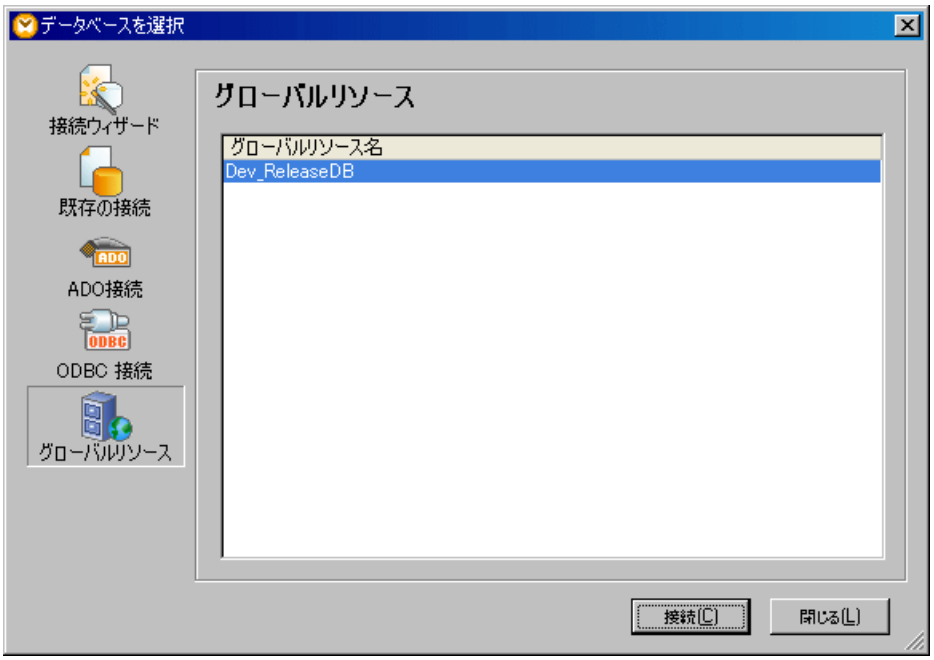
セットすることができるほか、スキップボタンをクリックすることで、決定を先送りすることができます。

データベースリソースの割り当て：

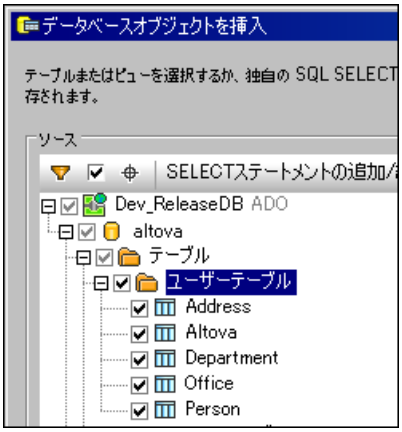
1. **Altova** データベースコンポーネントをダブルクリックして、変更ボタン（そして必要な場合は「グローバルリソース」ボタン）をクリックします。



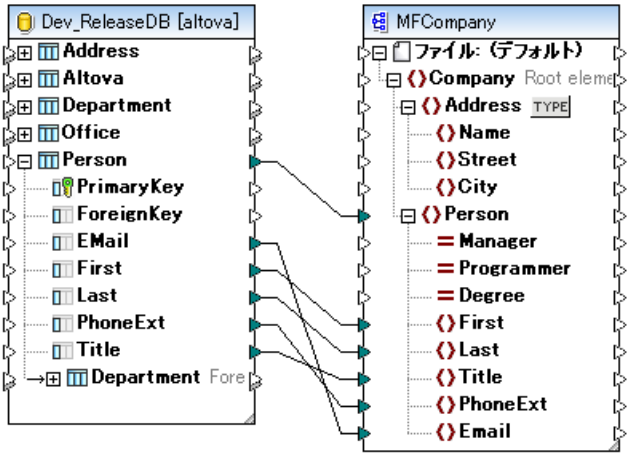
データベースの選択ダイアログボックスが表示されます。



2. **Dev_ReleaseDB** エントリーを選択して、接続をクリックします。

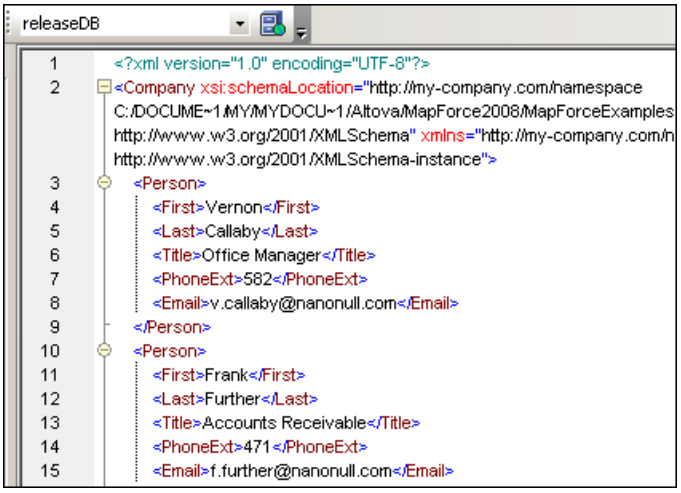


3. コンポーネントで使用したいテーブルを選択して、**「OK」**をクリックします。
リソースエイリアスがコンポーネントのヘッダーに**Dev_ReleaseDB [Altova]** というかたちで表示されます。



ランタイムにてデータベースリソースを変更する：

1. 出力タブをクリックしてマッピングの結果を確認します。.
ここで表示されるデータは ..\MapforceExamples フォルダーにあるデフォルトの Default データベースから得られていることに注意してください。
2. グローバルリソースコンボボックスをクリックして、**"releaseDB"** エントリーを選択します。
3. ダイアログが表示されるので**再ロード**をクリックします。
4. もう 1 度、出力ボタンをクリックして、マッピングの結果を確認します。



release データベースからのデータが表示されます。両者のデータベースに違いは無いため、この例で異なる結果が表示されることはありません。

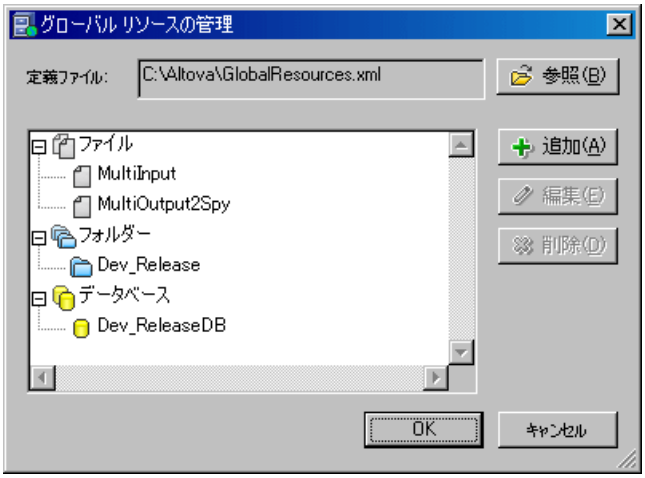
12.2.5 グローバルリソース - プロパティ

グローバルリソース XML ファイル

グローバルリソースの定義は XML ファイルに収められています。デフォルトでは、この XML ファイルは GlobalResources.xml という名前で、C:\Documents and Settings\<username>\My Documents\Altova\ フォルダー以下に収められています。このファイルは全ての Altova アプリケーションにおいて、デフォルトのグローバルリソース XML ファイルとしてセットされます。全てのアプリケーションがこのファイルを使用することにより、あるアプリケーションで定義されたグローバルリソースを、他の Altova アプリケーションでも使用することができるようになります。

ファイルの名前や場所の変更はいつでも行うことができ、複数のグローバルリソースXML ファイルを持つこともできます。各アプリケーションが同時に持つことができるアクティブなXML ファイルは 1 つだけで、そのアクティブなファイルに記述されている定義だけをアプリケーションで利用することができます。

グローバルリソースXML ファイルをアクティブにするには、定義ファイルフィールドにある参照ボタンをクリックして、使用するファイルをダイアログボックスにて選択します。



グローバルリソースの管理：追加、編集、削除

グローバルリソースダイアログでは、選択されたグローバルリソースXML ファイルに対してグローバルリソースの追加、編集、削除を行うことができます。グローバルリソースXML ファイルでは、ファイルとフォルダー、そしてデータベースという分類とともにエイリアスが管理されます。

グローバルリソースを追加する：

追加ボタンをクリックして、表示されるグローバルリソースダイアログボックスにてグローバルリソースの定義を行います。グローバルリソースの定義と保存を行うと、グローバルリソースが選択されたグローバルリソースXML ファイルのグローバル定義リストに追加されます。

グローバルリソースを追加する：

グローバルリソースを選択して、編集ボタンをクリックします。グローバルリソースダイアログが表示され、必要な変更を加えることができます。

グローバルリソースを削除する：

グローバルリソースを選択して、削除ボタンをクリックします。

アプリケーションワークフローにて結果を確認する：

呼び出し元のアプリケーション（例えばXMLSpy）により、他のアプリケーション（例えばMapForce）が呼び出される場合、グローバルリソースの管理ダイアログボックスにてビューボタンが利用可能になります。

ビューボタンをクリックすることで、現在選択されているグローバルリソースが呼び出し先のアプリケーションで得られる効果を確認することができます。サンプルについては グローバルリソース - アプリケーションワークフロー を参照ください。

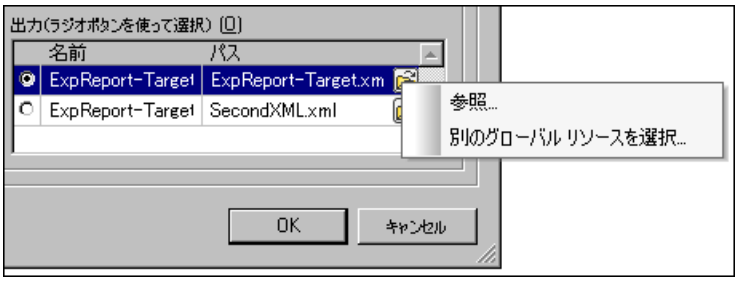
グローバルリソースの管理ダイアログボックスにて行われた変更を保存する：

グローバルリソースの追加、編集、または削除が完了した後には、グローバルリソースダイアログにある「**OK**」ボタンをクリックして、変更をグローバルリソースXML ファイルへ保存するようにしてください。

メモ：エイリアス名は、ファイルとフォルダー、データベースのセクション毎に一意な名前でなければなりません。しかし、セクション毎に同じ名前を使用することは可能で、例えば同じ名前のエイリアスをファイルとフォルダーセクションで作成することは可能です。

MapForce の変換結果をグローバルリソースとして選択

複数の出力が存在するMapForce の変換では、ラジオボタンをクリックすることで、グローバルリソースに対してどの出力ファイルが使用されるかを選択することができます



マッピングにより生成される**出力**ファイルは以下の方法により保存することができます：

- ポップアップにて別のグローバルリソースを選択エントリーを選択し**altova://file_resource/MF_output** という形式でパスが記述されます。出力は、グローバルリソースにより参照されているファイルへ記述されます。



- アイコンにより指定された (上のスクリーンショットではSecondXML.xml という名前の) ファイル。

  上にある方法のどちらも選択されていない場合、グローバルリソースが使用された時に、**一時的な** XML ファイルが作成されます。

ランタイムにてどのリソースを使用するか決定する

どのグローバルリソースを使用することができるか、またどのグローバルリソースが実際に使用されるのかを決定するには 2 つの方法があります：

- グローバルリソースダイアログにてアクティブなグローバルリソースXML ファイルを選択します。アクティブなグローバルリソースXML ファイルはいつでも変更することができ、アクティブになったファイルのグローバルリソース定義により、それまでの定義が更新されます。

  アクティブなグローバルリソースXML ファイルにより、(i) どのグローバルリソースを割り当てることができるのか、(ii) どのグローバルリソースを表示することができるのかが決定されます。 例えば、あるグローバルリソースXML ファイル内のグローバルリソースが割り当てられていたが、現在アクティブになっているグローバルリソースXML ファイルにて、同じ名前のグローバルリソースが存在しない場合、割り当てられたグローバルリソース (エイリアス) を表示することはできません)。

- メニューアイテム「ツール| アクティブな構成」またはグローバルリソースリソースツールバーにてアクティブな構成 を選択します。このコマンドを選択すると、全てのエイリアスにある構成が表示されます。

  構成を選択することで、その構成がアプリケーション全体でアクティブになります。この操作により、どこでグローバルリソース (またはエイリアス) が使用されようとも、アクティブな構成に割り当てられているリソースがロードされます。

  アクティブな構成は使用されている全てのエイリアスに対して割り当てられます。アクティブな構成に対して、対応するエイリアスが存在しない場合、そのエイリアスの**Default 構成**が使用されます。アクティブな構成はリソースの割り当てを行うにあたり重要ではなく、リソースが実際に使用されるかどうかが重要になります。

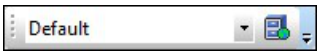
リソース / 構成の変更

リソースは、異なる構成名を選択することで切り替えることができます。以下にある 2 つの方法で実現することができます：

- メニューコマンド「ツール| アクティブな構成」へマウスカーソルを移動すると、グローバルリソースXML ファイルにて定義されている全ての構成が表示されます。目的の構成をサブメニューから選択します。



- グローバルリソースツールバーのコンボボックスにて、目的の構成を選択します。メニューコマンドから「ツール| カスタマイズ」を選択し、ツールバータブをクリックした後にグローバルリソースのチェックボックスを切り替えることで、グローバルリソースツールバーの表示を切り替えることができます。.



12.3 StyleVision を使用してマッピングの出力のスタイルを指定する

Altova StyleVision が使用中のコンピューターにインストールされていると仮定して、マッピングのターゲットコンポーネントが XML または XBRL である箇所は、マッピングの出力を HTML、RTF、PDF と Word 2007+ ドキュメントとしてプレビューし、保存することができます。StyleVision の Enterprise エディションが使用されている場合、プレビュー内でチャートを表示することができます。

マッピングが、これらのフォーマットのプレビューをサポートする場合、**出力** タブの横に使用することのできる追加タブが表示されます。例えば：



StyleVision プレビュータブ (MapForce Enterprise Edition)

以下に注意してください：

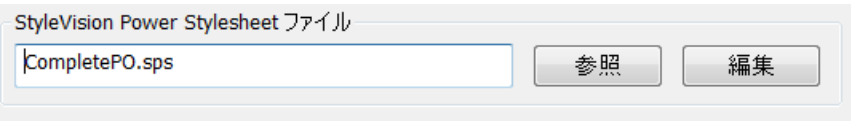
- StyleVision Professional がインストールされていると、**HTML** と **RTF** 出力をプレビューすることができます。StyleVision Enterprise を使用すると、**HTML**、**RTF**、**PDF** と **Word 2007+** 出力をプレビューすることができます。
- マッピングの出力を PDF としてプレビューするには、Java、Acrobat Reader、と FOP (Formatting Objects Processor) バージョン 0.93 または 1.0 .FOP が、StyleVision をインストールしないオプションが選択されていない限り、StyleVision と共にインストールされている必要があります。
- MapForce の 64 ビットエディションの場合、Word 2007+ と RTF プレビューは、埋め込まれていないアプリケーションとして開かれます。
- マッピングが ソースとターゲット (パススルーコンポーネント) としての役割を果たすコンポーネントを含む場合、コンポーネントの **プレビュー** ボタン が有効化されている場合のみ、StyleVision でのプレビューが可能になります。このようなマッピングの詳細に関しては、次を参照してください：チェーンマッピング.

マッピングからデータをこのようにプレビューするには、以下の必要条件が存在します：

- Altova StyleVision がコンピューターに、スタンドアロン、または Altova MissionKit の一部として、インストールされている必要があります。
- ターゲットコンポーネントには、関連付けられた StyleVision Power Stylesheet (SPS) ファイルが必要です。StyleVision を使用して、スタイルシートを作成、または編集することができます。MapForce で直接スタイルシートを編集、または、変更することはできませんが、StyleVision 内の MapForce から開くことはできます。スタイルシートの準備ができると、下に示されるようにターゲット MapForce コンポーネントに割り当てることができます。

ターゲットコンポーネントに StyleVision パワースタイルシートを割り当てる

1. StyleVision 内で、必要とするスタイルシートファイルを作成します。MapForce コンポーネントと同じ XML スキーマがソースとして使用されていることを確認してください。
2. MapForce 内で、ターゲット XML または XBRL コンポーネントを右クリックし、**プロパティ** を選択します。
3. コンポーネント設定ダイアログボックス内で、**StyleVision Power スタイルシートファイル** の横から、作成済みの スタイルシートファイルを選択します。



メモ： StyleVision Power スタイルシートファイルへのパスは、絶対または相対的であることができます。次を参照してください：相対的および絶対パスの使用。

StyleVision により生成された出力を保存する

StyleVision により生成されたHTML、PDF、RTF、またはWord 2007+ 出力 を、他のマッピングの保存と同様に保存することができます。具体的には、マッピングのプレビューを行うには、以下の内の1つをおこなってください：

- **生成された出力を保存する**（ ）ツールバーボタンをクリックします。
- **出力** メニューから、**出力ファイルを保存する**をクリックします。 .

Altova 製品のパッケージを使用して、HTML、PDF、RTF、Word 2007+ ファイルを自動 生成する

マッピングが（同じ、または、異なるマシン、または、プラットフォームで）HTML、PDF、RTF、およびWord 2007+ ファイルを自動的に生成する場合、（MapForce とStyleVision の機能性を拡張する個別にライセンスされるサーバー製品である）MapForce Server とStyleVision Server を使用して行うことができます。このシナリオの場合、各アプリケーションが以下の特別な役割を果たします：

- MapForce - （例えば、データベースからXML への）データ変換の入力と出力 を定義するマッピング（.mfd ファイル）のデザインを行うことができます。
- MapForceServer - マッピング（.mfx ファイル）をコマンドライン またはAPI から、（同じ、または、異なるオペレーティングシステム上で）実行可能なマッピングを実行します
- StyleVision - マッピングの出力をHTML、PDF、RTF、Word 2007+に出力するために必要な、スタイルシート（.sps ファイル）をデザインすることができます。
- StyleVision Server - マッピングの出力をターゲットフォーマットに変換する .sps スタイルシートを作成します。これは、コマンドライン、または、API（ 同じ、または、異なるオペレーティングシステム）から行うことができます。
- （個別のライセンスを必要とする）StyleVision Server とMapForce Server は、任意でFlowForce Server の管理下で作動します。このシナリオの場合、MapForce マッピング とStyleVision 変換は、予定通り、実行、トリガー、または、需要に応じて実行することができ、完全に自動化することができます。

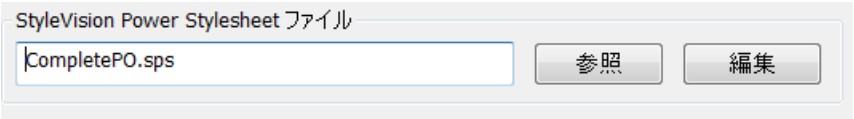
12.3.1 StyleVision スタイルシートを使用したマッピングの例

MapForce サンプルフォルダー（**<Documents>\Altova\MapForce2017\MapForceExamples**）内のマッピングの多くは、ターゲットコンポーネントに割り当てられているStyleVision Power Stylesheets (.sps ファイル）を含んでいます。この場合、マッピングはをStyleVision プレビュータブ含んでいます。

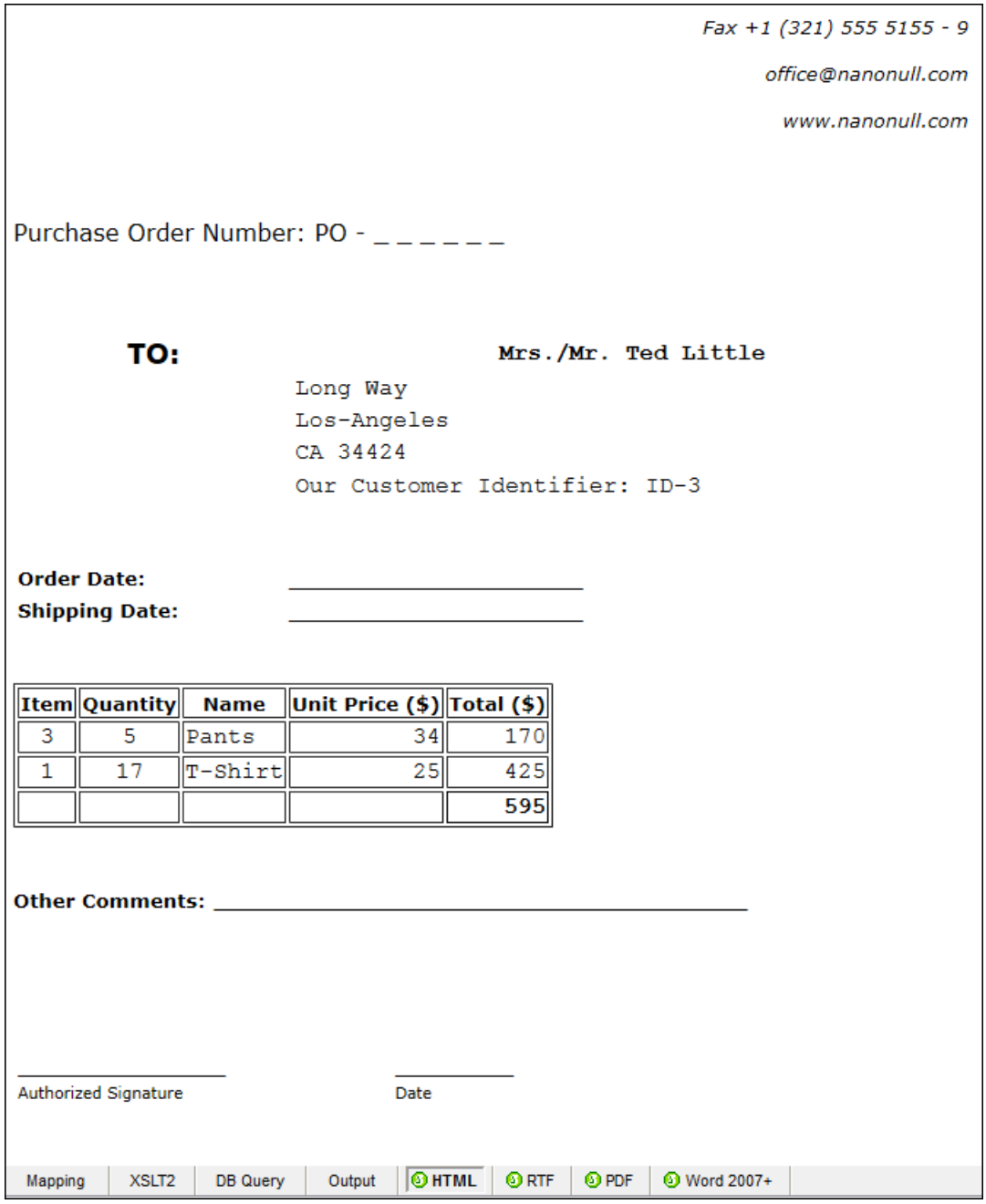


StyleVision プレビュータブ (MapForce Enterprise Edition)

次のパスで見つけることのできる**CompletePO.mfd** はスタイルシートを使用したマッピングの一例です：**<Documents>\Altova\MapForce2017\MapForceExamples\CompletePO.mfd**。このマッピングは、XML フォーマットで注文を表示します。ターゲットコンポーネントを右クリックして、**プロパティ**を選択すると、sps ファイルが割り当てられていることに気が付きます。



HTML フォーマットで表示される出力データを確認するために**出力** タブをクリックします。



Fax +1 (321) 555 5155 - 9

office@nanonull.com

www.nanonull.com

Purchase Order Number: PO - _ _ _ _ _ _

TO: **Mrs./Mr. Ted Little**
Long Way
Los-Angeles
CA 34424
Our Customer Identifier: ID-3

Order Date: _____
Shipping Date: _____

Item	Quantity	Name	Unit Price ($)	Total ($)
3	5	Pants	34	170
1	17	T-Shirt	25	425
				595

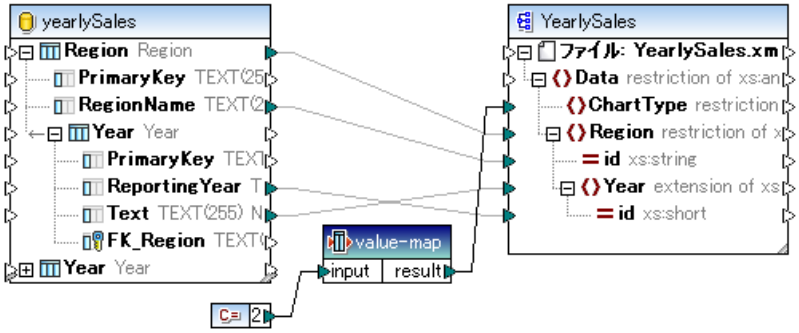
Other Comments: _____

_____ _____
Authorized Signature Date

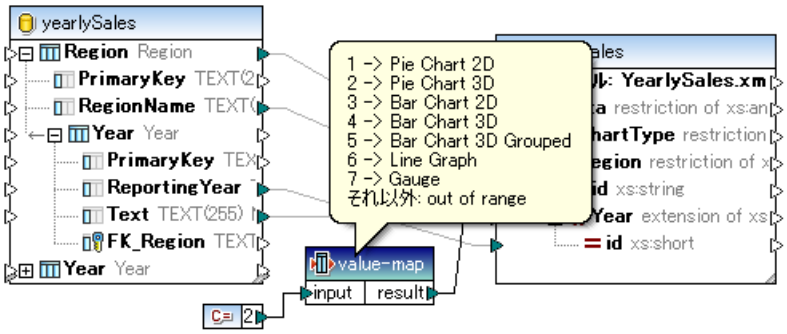
Mapping | XSLT2 | DB Query | Output | HTML | RTF | PDF | Word 2007+

CompletePO.mfd のHTML プレビュー(MapForce Enterprise edition)

他のサンプルは**YearlySales.mfd** です。このマッピングは、以下のパスで見つけることができます :**<Documents> \Altova\MapForce2017\MapForceExamples\Tutorial\YearlySales.mfd**。



このマッピングに割り当てられているスタイルシートは、StyleVision を使用してデザインされており、`ChartType` 要素の値を変更して、チャートの型の管理が可能することができます。これにより、マッピングから直接チャートの型を変更することができます。具体的な、値の定数のデフォルトの値を 1 から 7 に変更することができます。 **value-map** コンポーネントにマウスをポイントすると、使用可能な値が表示されます。



出力内での 3D 円グラフを生成する定数のデフォルトの値は、 "2" です。他の種類のチャートを表示するには、他の許可されている値に変更し、結果を確認するために **出力** タブをクリックします。

12.4 マッピングドキュメントの生成とカスタム化

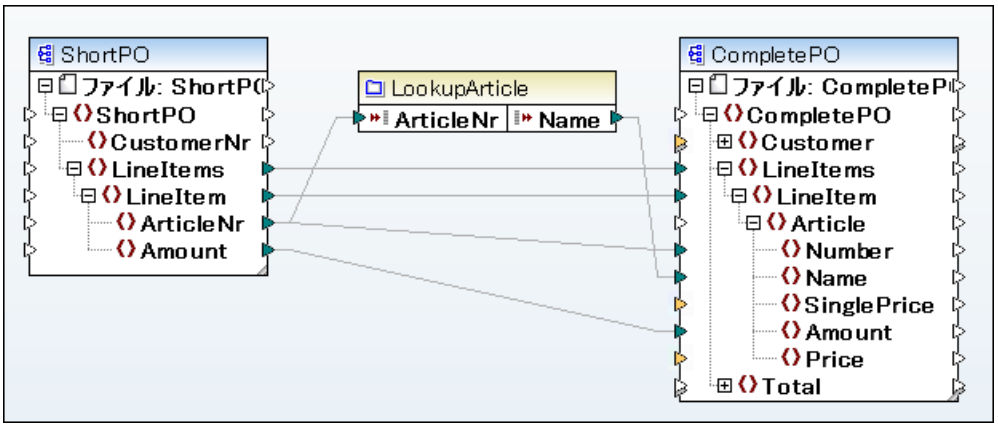
ドキュメンテーションの生成コマンドにより、マッピングに関する詳細なドキュメンテーションをHTML、MS Word、RTF、またはPDF 形式で生成することができます。このコマンドにより生成されたドキュメンテーションはAltova からの許可を得ること無く自由に改変、使用することができます。

ドキュメンテーションは、ドキュメンテーションの生成ダイアログボックスにて選択したコンポーネントに対して生成されます。固定デザインまたはStyleVision のSPS ファイルをデザインとして使用することができます。StyleVision SPS を使用することで、生成されるドキュメンテーションのデザインをカスタマイズすることができます。カスタマイズの方法については（ユーザー定義デザインのセクションを参照ください）。

メモ： SPS ファイルによりマッピングドキュメンテーションを生成するには、お使いのコンピューターにStyleVision がインストールされている必要があります。画面への出力では、関連する要素が通常ハイパーリンクされており、コンポーネント間のナビゲーションを行うことができます。

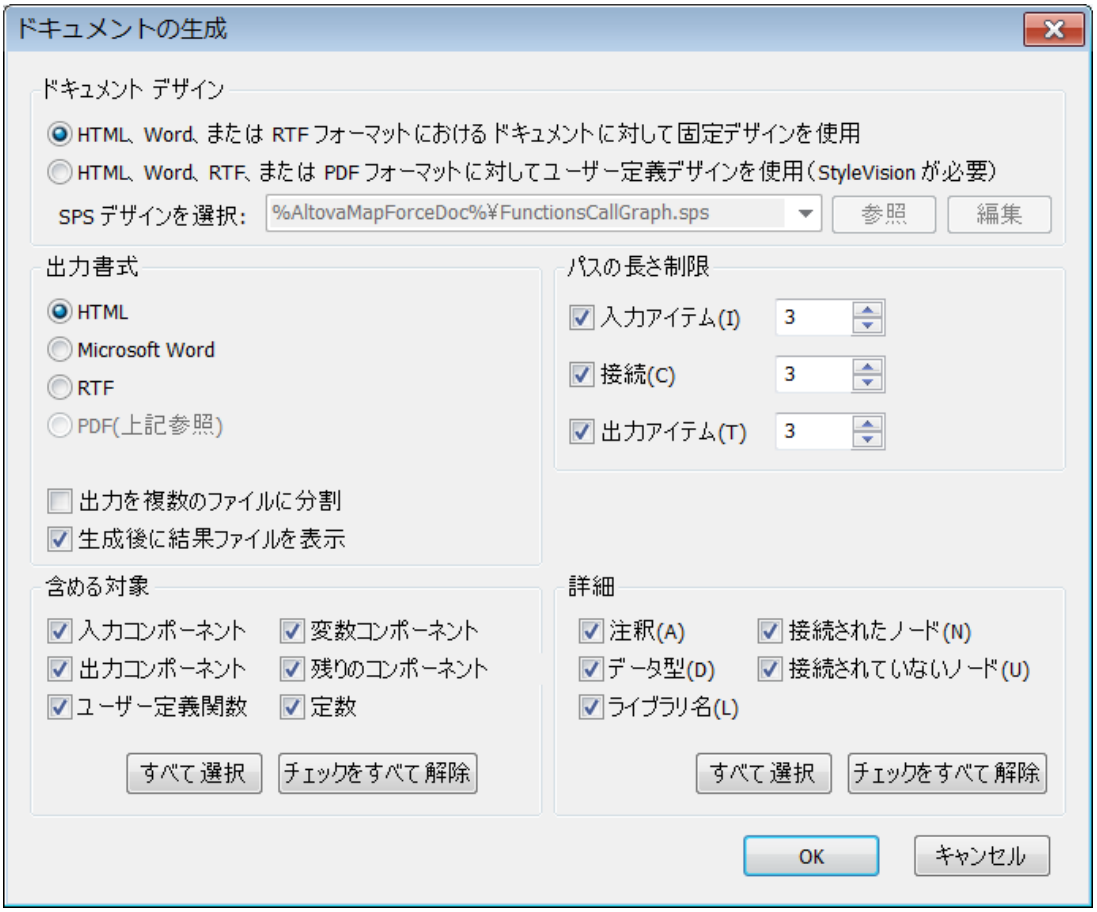
MS Word フォーマットにてドキュメンテーションの生成を行うには、MS Word （バージョン 2000以降）がインストールされている必要があります

以下のスクリーンショットでは、**..\MapForceExamples** フォルダー以下の**Lookup-standard.mfd** ファイルが示されます。



マッピングファイル（例：Lookup-standard.mfd）が開かれている状態で以下の操作を行います：

- メニューオプションから「ファイル| ドキュメントの生成」を選択します。
  この操作によりドキュメントの生成ダイアログボックスが表示されます。デフォルトのダイアログボックス設定を以下のスクリーンショットに示します。



ドキュメンテーションデザイン

- 「固定デザインを使用」を選択することで、内蔵のドキュメンテーションテンプレートを使用します。
- 「ユーザー定義デザインを使用」を選択することで、StyleVision により作成された StyleVision Power Stylesheet を使用します。"..**My Documents\Altova\MapForce2013\Documentation\MapForce**\" 以下に収められている SPS ファイルを選択することができます。
- **参照**ボタンをクリックして定義済みの SPS ファイルを選択することができます。
- **編集**ボタンをクリックすると StyleVision が起動し、選択された SPS が StyleVision ウィンドウにて開かれます。

以下にある定義済み SPS スタイルシートが .MapForce2013\Documentation\MapForce フォルダーに収められています:

- **FunctionCallGraph.sps** - メインマッピングの呼び出しグラフならびにユーザー定義関数が表示されます。
- **FunctionsUsedBy.sps** - マッピングにて、どの関数が直接または間接的に使用されているか表示します。
- **ImpactAnalysis.sps** - 全てのソースならびにターゲットノードを、ターゲットノードまでのルートとともに表示します。
- **OverallDocumentation.sps** - 全てのノード、接続、関数、そしてターゲットノードを表示します。このオプションは最も詳細で、内蔵の「固定デザイン」出力を使った出力と同一の出力を得ることができます。

出力書式

- 出力フォーマットをHTML、Microsoft Word、RTF、またはPDF から選択します。

  固定デザインを使用すると、Microsoft Word ドキュメントは **.doc** ファイル拡張子で作成され、StyleVision SPS を使った生成では **.docx** ファイル拡張子にてファイルが作成されます。

  PDF 出力フォーマットは、StyleVision SPS を使ったドキュメンテーションの生成でのみ有効になります

- **出力を複数のファイルに分割**オプションを選択すると、入力、出力、定数コンポーネント、ライブラリから得られるユーザー定義関数のドキュメンテーションが分割されます。固定デザインでは、複数のドキュメント間においてリンクが自動的に作成されます。

- **生成時にファイルを表示**オプションは全ての出力オプションに対して選択可能です。有効になっている場合、生成されたファイルがデフォルトのブラウザー (HTML 出力)、MS Word (MS Word 出力)、.rtf ファイルのデフォルトアプリケーション (RTF 出力) にて開かれます。

パスの長さ制限

アイテムに表示される「パス」の最大長さを指定することができます。

- 例えば、 .\ShortPO\LineItems\LineItem は、デフォルトで設定されている最大長さ 3 となります。

含める対象

- ドキュメンテーションに表示されるコンポーネントを指定することができます。

詳細

ドキュメンテーションに表示される詳細情報を指定することができます。

- ライブラリ名を選択することで、関数に「core」プレフィックスが挿入されます。
- 接続されたノードならびに接続されていないノードの両方をドキュメンテーションに含めることができます。

メモ:
全てチェック全てクリアボタンにより、そのグループにある全てのチェックボックスにチェックを入れる/チェックを外すことができます。

上の示されているデフォルトの設定にて「**OK**」をクリックすると、出力ファイルの保存先とその名前を選択するよう促されます。固定デザインにより生成されたドキュメンテーションの一部を以下に示します。これは単一の出力ファイルであることに留意してください。

このテーブルではソースコンポーネント**から**ターゲットコンポーネントへの接続が示されます。

入力 **ShortPO** (ShortPO.xsd)

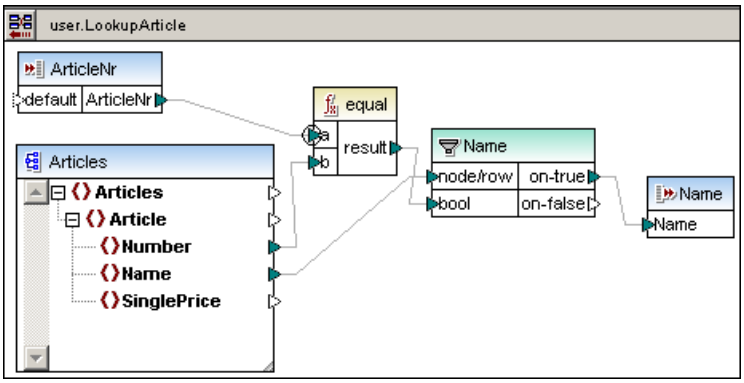
ノード	接続	
ファイル: ShortPO.xml 型: string		
ShortPO 型: restriction of xs:anyType [0..1]		
ShortPO/CustomerNr 型: xs:integer		
ShortPO/LineItems 型: restriction of xs:anyType	*直接*	CompletePO/LineItems 型: restriction of xs:anyType
ShortPO/LineItems/LineItem 型: restriction of xs:anyType [1..∞]	*直接*	CompletePO/LineItems/LineItem 型: restriction of xs:anyType [1..∞]
.../LineItems/LineItem/ArticleNr 型: xs:integer	*直接*	.../LineItem/Article/Number 型: xs:integer
	user.LookupArticle => ArticleNr | Name =>	.../LineItem/Article/Name 型: xs:string
.../LineItems/LineItem/Amount 型: xs:integer	*直接*	.../LineItem/Article/Amount 型: xs:integer

コンポーネントがドキュメント化される順番は、入力、出力、定数、ユーザー定義関数、そしてライブラリ関数となります。

例 :**入力コンポーネント** ShortPO:

- 最初の 2 つのアイテムである ShortPO と ShortPO/CustomerNr はターゲット内にあるどのアイテムにも接続されておらず、カラムが空となっています。
- **ShortPO/LineItems** はターゲットの CompletePO/**LineItems** へ直接接続されています。
- /LineItems/LineItem/**ArticleNr** には 2 つの接続が行われています :
  - ターゲットの /LineItem/Article/**Number** へ直接接続されています。
  - 入力パラメーターとなる **LookupArticle** という名前のユーザー定義関数へ ArticleNr が接続されており、Name という名前の出力パラメーターがターゲットへ接続されています。

  ユーザー定義関数の内容を以下に示します。



出力コンポーネントの CompletePO ：このテーブルではソースコンポーネントからターゲットコンポーネント**への**接続が表示されます

出力 **CompletePO** (CompletePO.xsd)

接続		ノード
		ファイル: CompletePO.xml 型: string
		CompletePO 型: restriction of xs:anyType [0..1]
		CompletePO/Customer 型: restriction of xs:anyType [0..1]
ShortPO/LineItems 型: restriction of xs:anyType [0..1]	*直接*	CompletePO/LineItems 型: restriction of xs:anyType [0..1]
ShortPO/LineItems/LineItem 型: restriction of xs:anyType [0..∞]	*直接*	CompletePO/LineItems/LineItem 型: restriction of xs:anyType [0..∞]
		.../LineItems/LineItem/Article 型: extension of ArticleType [0..1]
.../LineItems/LineItem/ArticleNr 型: xs:integer [0..1]	*直接*	.../LineItem/Article/Number 型: xs:integer [0..1]
.../LineItems/LineItem/ArticleNr 型: xs:integer [0..1]	**user.LookupArticle** => ArticleNr | Name =>	.../LineItem/Article/Name 型: xs:string [0..1]

- 最初の 2 つのアイテムである CompletePO と CompletePO/Customer はソースコンポーネント内にあるどのアイテムにも接続されておらず、カラムが空となっています。
- **CompletePO**/LineItems はソースコンポーネントの **ShortPO**/LineItems へ直接接続されています。
- LineItem/Article/Name は LineItems/LineItem/ArticleNr をソースアイテムとするユーザー定義関数の

LookupArticle へ接続されています。

ユーザー定義関数

user.LookupArticle

入力 (必須) core.ArticleNr

ノード	接続	
ArticleNr 型: string	core.equal => a | result => core.filter => bool | on-true =>	core.output => Name 型: string

入力 **Articles** (Articles.xsd)

ノード	接続	
ファイル: Articles.xml 型: string		
Articles 型: restriction of xs:anyType [0..1]		
Articles/Article 型: ArticleType [0..∞]		
Articles/Article/Number 型: xs:integer [0..1]	core.equal => b | result => core.filter => bool | on-true =>	core.output => Name 型: string
Articles/Article/Name 型: xs:string [0..1]	core.filter => node/row | on-true =>	core.output => Name 型: string
Articles/Article/SinglePrice 型: xs:decimal [0..1]		

12.4.1 定義済みの SPS スタイルシート

関数の呼び出しグラフ -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

使用される関数 -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

影響分析 -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

全体的なドキュメント -PersonListByBranchOffice.mfd

Mapping **PersonListByBranchOffice.mfd**

Input **core.OfficeName**

Nodes	Connections	
OfficeName Type: string	**core.equal => a | result =>** **core.filter => bool | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
	user.LookupPerson => **Office_Name | result =>**	PersonList/Person/Details Type: xs:string [0..1]

Input **BranchOffices** (BranchOffices.xsd)

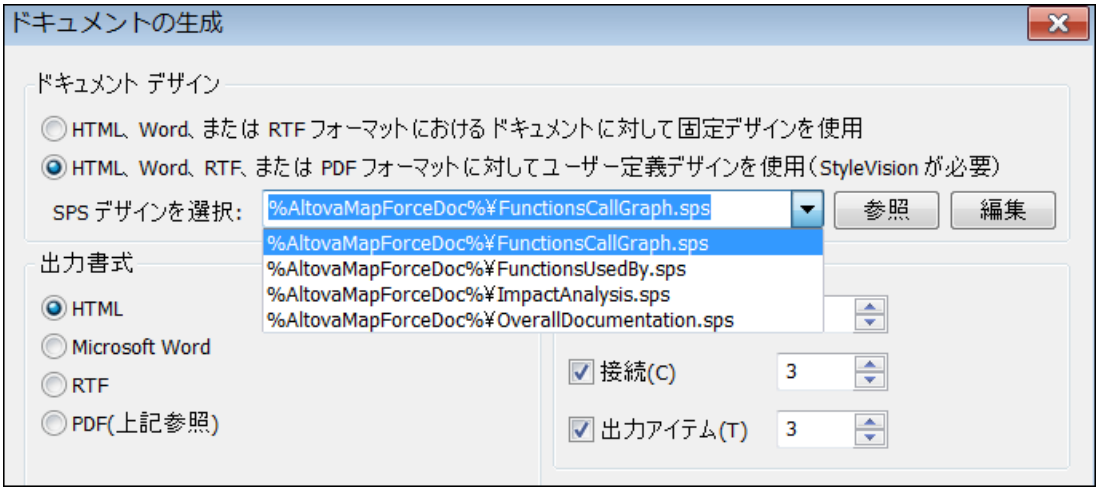
Nodes	Connections	
File: BranchOffices.xml Type: string		
BranchOffices Type: restriction of xs:anyType [0..1]		
BranchOffices/Name Type: restriction of xs:string		
BranchOffices/Office Type: restriction of xs:anyType [0..∞]	**core.filter => node/row | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
BranchOffices/Office/Name Type: restriction of xs:string	**core.equal => b | result =>** **core.filter => bool | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons

12.4.2 カスタムデザイン

固定デザインの代わりに、MapForce ドキュメンテーション用にカスタマイズされたデザインを作成することもできます。カスタマイズされたデザインは StyleVision SPS にて作成されます。MapForce には 4 つの定義済み SPS スタイルシートが収められています。マッピングプロジェクトのドキュメントを参照ください。

MapForce のドキュメンテーションで使用する SPS を指定する

ドキュメンテーションの生成に使用する SPS は、**「ファイル」| 「ドキュメンテーションの生成」**からアクセスすることのできるドキュメンテーションの生成ダイアログにて指定することができます。「ユーザー定義デザインを使用」ラジオボタンを選択し、ドロップダウンをクリックすることで目的のファイルを選択します。デフォルトの選択は **OverallDocumentation** エントリーとなっています。



これらの定義済み SPS ファイルは .MapForce2013\Documentation\MapForce フォルダーに収められています。

メモ:
SPS によるドキュメンテーションの生成を行うには、お使いのコンピューターに StyleVision がインストールされている必要があります。

SPS を作成する

StyleVision Power Stylesheet (SPS) は Altova StyleVision により作成されます。MapForce のドキュメンテーションを生成するための SPS は MapForce ドキュメンテーションを含む XML ドキュメントの構造を指定する XML スキーマをベースにしている必要があります。

スキーマは MapForceDocumentation.xsd という名前で、..\My Documents\Altova\MapForce2013\Documentation フォルダーに収められています。

StyleVision にて SPS デザインを作成する際、MapForceDocumentation.xsd スキーマから得られたノードがデザインに配置され、スタイルやプロパティに割り当てられます。MapForceDocumentation.xsd は、配置されているフォルダーの親フォルダーにある Documentation.xsd ファイルを**インクルード**していることに注意してください。

リンクやイメージといったその他のコンポーネントを SPS デザインへ含めることもできます。StyleVision にて SPS デザインを作成する方法の詳細については、StyleVision ユーザーマニュアルを参照ください。

SPS によりマッピングドキュメンテーションを生成することで、ドキュメンテーションのデザインを細部まで調整することができるようになります。SPS を使用することで PDF フォーマットによるドキュメンテーションの出力も可能になります。PDF によるドキュメンテーションの生成は固定デザインでも使用することができます。

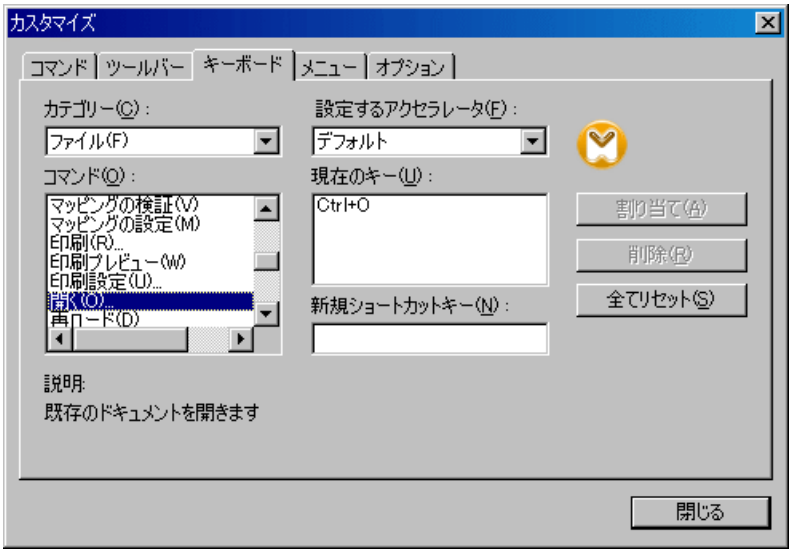
12.5 キーボードのショットカットのカスタム化

MapForce 内のキーボードのショットカットを以下のように定義または変更することができます :

1. **「ツール」** メニューから **「カスタマイズ」** をクリックします。
2. **「キーボード」** タブをクリックします。

コマンドに新規ショートカットを割り当てる :

1. **「ツール」 「カスタマイ」** コマンドを選択して、キーボードタブをクリックします。
2. **カテゴリー** コンボボックスをクリックして、メニュー名を選択します。
3. コマンドリストボックス内で新しいショートカットを割り当てる**コマンド**を選択します。
4. **新規ショートカットキー**をクリックします : テキストボックスから、コマンドを有効化するショートカットキーを押します。



ショートカットはテキストボックス内にすぐに表示されます。ショートカットが以前に割り当てられている場合、その関数がテキストボックスの下に表示されます。

5. **割り当て** ボタンをクリックして、ショートカットを割り当てます。
現在のキーリストボックスにショートカットが表示されます。
(新規ショートカットキーテキストボックス内のエントリを**クリア**するには、コントロールキーを押します **CTRL**、**ALT** または **SHIFT**)。

ショートカットを元に戻す、または削除する :

1. 現在のキーリストボックス内から削除するショートカットをクリックします。
2. **削除** ボタンをクリックします。
3. **閉じる** ボタンクリックして、確認します。

メモ : 設定するアクセラレーターには現在関数が存在しません

現在キーボードに割り当てられているショートカットは、以下のとおりです :

F1	ヘルプメニュー
F2	次のブックマークへ (出力ウィンドウにて)
F3	次を検索
F10	メニューバーをアクティベート
Num +	現在のアイテムノードを展開

Num -	アイテムノードを縮退
Num *	現在のアイテムノード以下を全て展開
CTRL + TAB	開かれているマッピング間の切り替え
CTRL + F6	開かれているウィンドウをサイクル
CTRL + F4	アクティブなマッピングドキュメントを閉じる
Alt + F4	MapForce を閉じる
Alt + F, F, 1	最後に閉じられたファイルを開く
Alt + F, T, 1	最後に閉じられたプロジェクトを開く
CTRL + N	新規ファイル
CTRL + O	ファイルを開く
CTRL + S	ファイルの保存
CTRL + P	ファイルの印刷
CTRL + A	すべて選択
CTRL + X	切り取り
CTRL + C	コピー
CTRL + V	貼り付け
CTRL + Z	元に戻す
CTRL + Y	やり直し
Del	コンポーネントを削除（プロンプト付き）
Shift + Del	コンポーネントを削除（プロンプトなし）
CTRL + F	検索
F3	次を検索
Shift + F3	前を検索

矢印キー

（上 / 下）	コンポーネント内にあるアイテムを選択
Esc	編集の破棄 / ダイアログボックスを閉じる
Return	選択の確定

出力ウィンドウホットキー

CTRL + F2	ブックマークの挿入 削除
F2	次のブックマークへ
SHIFT + F2	前のブックマークへ
CTRL + SHIFT + F2	全てのブックマークを削除

ズーム（倍率）ホットキー

CTRL + 上向きマウスホイール	ズームイン
CTRL + 下向きマウスホイール	ズームアウト
CTRL + 0 (ゼロ)	倍率のリセット

12.6 カタログファイル

MapForce は OASIS XML カタログ機能のサブセットをサポートしています。MapForce のカタログ機能により、広く使われているスキーマ (またはスタイルシートなどその他のファイル) をローカルユーザーフォルダーから取得することができます。これにより全体的な処理スピードが向上し、(ネットワークに接続していない状態でも) ユーザーはオフラインで作業することができるようになり、(URI の変更はカタログファイル内部に限定されるので) ドキュメントのポータビリティーが向上します。

MapForce のカタログ機能は以下のように動作します。

RootCatalog.xml

MapForce が開始されると、(以下に示されるような構造を持った) RootCatalog.xml というファイルがロードされ、ファイル内部に記述されているカタログファイルが検索されます。このファイルを修正して、検索するカタログファイルを増やすこともできます (各カタログファイルは nextCatalog 要素にて記述されます)。各カタログファイル内に記述されている URI が、ファイル内で指定されたマッピングに従って解決されます。

```
<?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>
```

上で示されるファイルの内容にて、%AltovaCommonFolder% という変数で指定されている スキーマならびに XBRL フォルダー内部に catalog.xml という名前のカタログファイルがあることに注目してください (%AltovaCommonFolder% 変数の値については以下を参照ください)。

Altova 共通フォルダー以下にあるカタログファイルにより、SVG や WSDL といった よく使われる定義済みのスキーマや XBRL タクソノミのパブリックならびにシステム識別子が、ローカルにコピーされた対応するスキーマを指し示す URI にマッピングされます。これらのスキーマは MapForce がインストールされる際に Altova 共通フォルダーにインストールされます。エラーが発生する可能性があるので、これらファイル内のマッピングを複製しないようご注意してください。

CoreCatalog.xml、CustomCatalog.xml、そして Catalog.xml

上で記されている RootCatalog.xml にて、CoreCatalog.xml と CustomCatalog.xml が検索のために使用されていることに注目してください:

- `CoreCatalog.xml` には `Altova` 共通フォルダーを指し示すために使われる `Altova` 独自のマッピングが含まれています。
- `CustomCatalog.xml` はマッピングを作成するためのスケルトンファイルです。`Altova` 共通フォルダー以下のカタログファイルで指定されていないスキーマファイルへのマッピングを `CustomCatalog.xml` に追加することができます。`OASIS` カタログ機能によりサポートされている要素を使用してください (以下を参照)。
- Altova 共通フォルダーには数多くの Catalog.xml ファイルが格納されています。各ファイルは Altova 共通フォルダー以下のスキーマフォルダーまたは XBRL タクソノミフォルダーに格納され、ローカルにコピーされたパブリックならびにシステム識別子のスキーマを指し示す URI のマッピングを行ないます。

カタログファイルとスキーマの場所

RootCatalog.xml ならびに CoreCatalog.xml は MapForce アプリケーションフォルダー以下にインストールされます。CustomCatalog.xml は MyDocuments\Altova\MapForce フォルダー以下に納されています。catalog.xml ファイルは各スキーマフォルダー以下にあり、これらスキーマフォルダーは %AltovaCommonFolder%\Schema ならびに %AltovaCommonFolder%\XBRL 以下に収められています。

シェル環境変数と Altova 変数

nextCatalog 要素ではシェル環境変数が使用され、システムに関係する場所へのパスが参照されます（上の RootCatalog.xml の内容を参照）。以下のシェル環境変数がサポートされます：

%AltovaCommonFolder%	C:\Program Files\Altova\CommonMapForce
%DesktopFolder%	現在のユーザーのデスクトップフォルダーへのフルパス
%ProgrammenuFolder%	現在のユーザーのプログラムメニューフォルダーへのフルパス
%StartMenuFolder%	現在のユーザーのスタートメニューフォルダーへのフルパス
%StartUpFolder%	現在のユーザーのスタートアップフォルダーへのフルパス
%templateFolder%	現在のユーザーのテンプレートフォルダーへのフルパス
%AdminToolsFolder%	現在のユーザーの管理ツールが収められているシステムディレクトリへのへのフルパス
%AppDataFolder%	現在のユーザーのアプリケーションデータフォルダーへのフルパス
%CommonAppDataFolder%	全てのユーザーのアプリケーションデータが収められているファイルディレクトリへのフルパス
%FavoritesFolder%	現在のユーザーのお気に入りフォルダーへのフルパス
%PersonalFolder%	現在のユーザーの個人用フォルダーへのフルパス
%SendToFolder%	現在のユーザーの SentTo フォルダーへのフルパス
%FontsFolder%	システムフォントフォルダーへのフルパス
%ProgramFilesFolder%	現在のユーザーの Program Files フォルダーへのフルパス
%CommonFilesFolder%	現在のユーザーの Common Files フォルダーへのフルパス
%WindowsFolder%	現在のユーザーの Windows フォルダーへのフルパス
%SystemFolder%	現在のユーザーのシステムフォルダーへのフルパス
%CommonAppDataFolder%	全てのユーザーのアプリケーションが収められているファイルディレクトリへのフルパス

%LocalAppDataFolder%	ローカルアプリケーションのデータリポジトリとして使用されるファイルシステムディレクトリへのフルパス
%MyPicturesFolder%	マイピクチャフォルダーへのフルパス

カタログの動作

DTD が呼び出された際のリダイレクトにカタログは使用されます。カタログファイル内に記述されているマッピング情報を使用することで、パブリックまたはシステム識別子をローカルの URI にマッピングします。例えば XML ファイル内で宣言されている DOCTYPE 宣言が読まれたとき、カタログファイルマッピングにより PUBLIC または SYSTEM 識別子がローカルにあるリソースを参照します。

よく使われるスキーマや PUBLIC 識別子は通常定義が定まっており、ローカルにコピーされている URI をカタログファイルが指し示すだけで事足りてしまいます。XML ドキュメントが解析されると内部にある PUBLIC 識別子が読み込まれます。カタログファイル内に識別子が見つかった場合、カタログファイル内の対応する URL が検索され、その場所からスキーマが読み込まれます。例えば、以下の SVG ファイルが MapForce で開かれた場合：

```
<?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>
```

このドキュメントが読み込まれ、PUBLIC 識別子のカタログが検索されます。例えば、カタログファイルに以下のような記述があるとしましょう：

```
<catalog>
  ...
    <public publicId="-//W3C//DTD SVG 1.1//EN" uri="schemas/svg/svg11.dtd"/>
  ...
</catalog>
```

これで PUBLIC 識別子に対するマッチが見つかりました。SVG DTD は schemas/svg/svg11.dtd にリダイレクトされ（パスはカタログファイルからの相対パスになります）、このローカルファイルが DTD として使用されます。カタログ内に対応するマッピングが存在しない場合、XML ドキュメントに記述されている URL （上の例の場合だと http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd）が使用されます。

MapForce によりサポートされるカタログサブセット

CustomeCatalog.xml （または MapForce により読み込まれる他のカタログファイル）内でエントリを作成する際、以下に示される OASIS カタログ仕様の要素だけを使うようにしてください。以下に各要素と属性の説明を示します。更に詳しい情報については、XML カタログ仕様（英文）を参照ください。各要素は xml:base 属性を持ち、要素のベース URI を指定できることに注意してください。

- `<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"/>`

始どのスタイルシートの様に、パブリックな識別子が存在しない場合、system 要素を使うことでシステム識別子をURL に直接マッピングすることができます。また、uri 要素によりURI を別のURI にマッピングすることもできます。rewriteURI と rewriteSystem 要素によりURI またはシステム識別子の最初の文字列を書き換えることができます。これによりファイルパスの最初の部分を置き換えて、別のディレクトリにターゲットを変更することができます。これら要素に関する詳しい情報については XML カタログ仕様 (英文) を参照ください。

スキーマに従ったファイル拡張子とインテリジェントな編集

指定したスキーマのルールに準拠するようMapForce のインテリジェントな編集機能が適用された場合、カタログファイルを使うことで、ドキュメントに特定の拡張子を指定することができます。例えば独自のファイル拡張子 .myhtml を作成して、HTML DTD に従ったファイルの検証を行ないたい場合、以下の要素をCustomCatalog.xml 内にある <catalog> 要素の子要素として追加することで、この拡張子を持つファイルのインテリジェント編集を行うことができます。

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

これによりMapForce 内で .myhtml ファイルを編集する際のインテリジェント編集 (自動補完、入力ヘルパーなど) が XHTML 1.0 Transition DTD に従った形で使えるようになります。似たようなエントリを含んでいる %AltovaCommonFolder%\Schemas\xhtml フォルダー以下の catalog.xml ファイルも参照ください。

XML スキーマとカタログ

XML スキーマ情報はMapForce に内蔵されており、XML スキーマドキュメントの妥当性は内部にある情報をベースにチェックされます。そのため、XML スキーマではいかなる参照も発生しません。

%AltovaCommonFolder%\Schemas\schema フォルダー以下にある catalog.xml ファイルには、古い XML スキーマ仕様の DTD に対する参照が記述されています。このファイルから参照されるスキーマファイルを使って XML スキーマドキュメントの検証を行うべきではありません。これらのファイルは、以前の勧告に従ったドキュメントを作成する際に、MapForce の入力ヘルパーにて該当する編集情報を表示させるためだけに使用されます。

更に詳しい情報

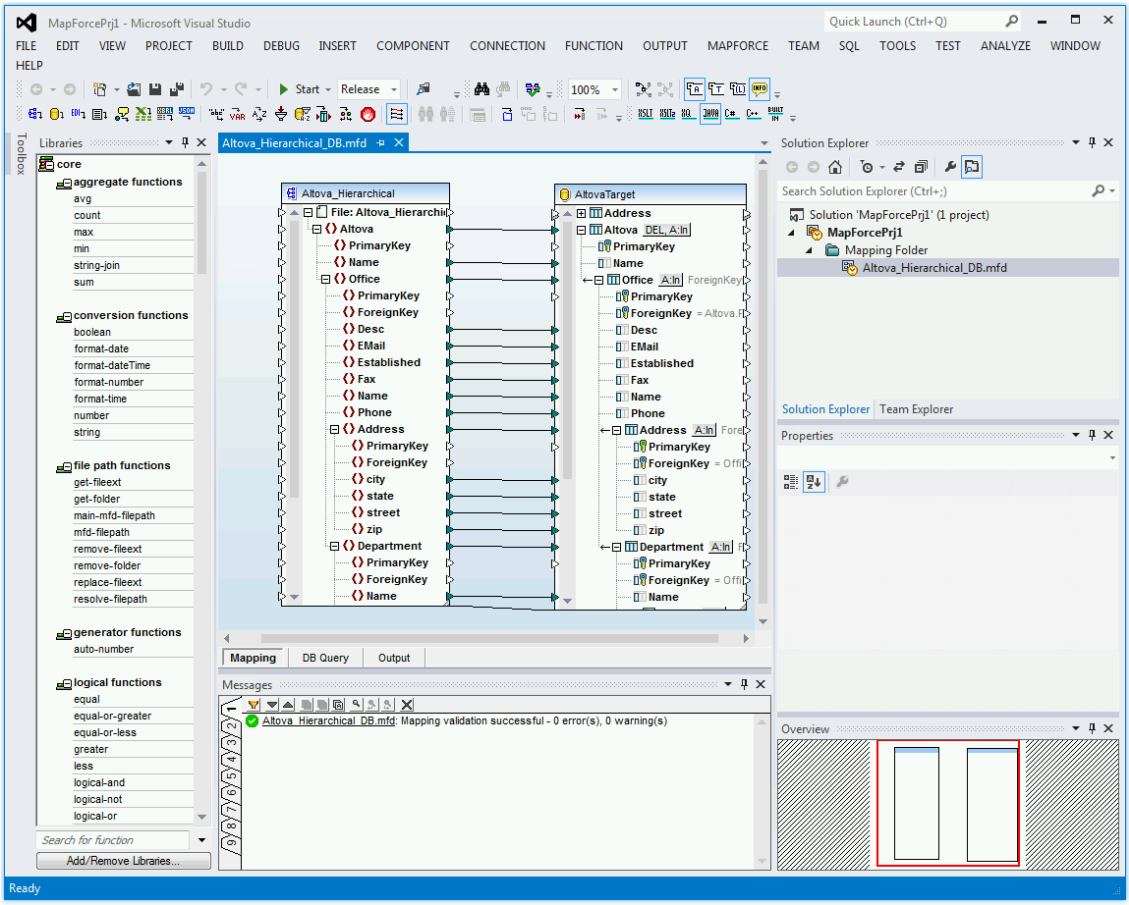
カタログに関する更に詳しい情報は XML カタログ仕様 (英文) を参照ください。

Chapter 13

Visual Studio のための MapForce プラグイン

13 Visual Studio のための MapForce プラグイン

MapForce 2017 を Microsoft Visual Studio バージョン 2005/2008/2010/2012/2013/2015/2017 に統合することができます。これはMapForce のマッピング機能とVisual Studio の開発環境と2つの世界をつなぎます。MapForce プラグインが有効化されると、マッピング プロジェクトとファイルを直接 Visual Studio から作成することができます。MapForce のスタンドアロンのバージョンと同様に、メニューとツールバーを含む、MapForce オプションをカスタム化することができます。



MapForce Enterprise Edition プラグイン (Visual Studio 2012)

このセクションは、以下のトピックを含みます：

- プラグインの有効化
- マッピングとプロジェクトと作業
- 共通メニューと関数へのアクセス

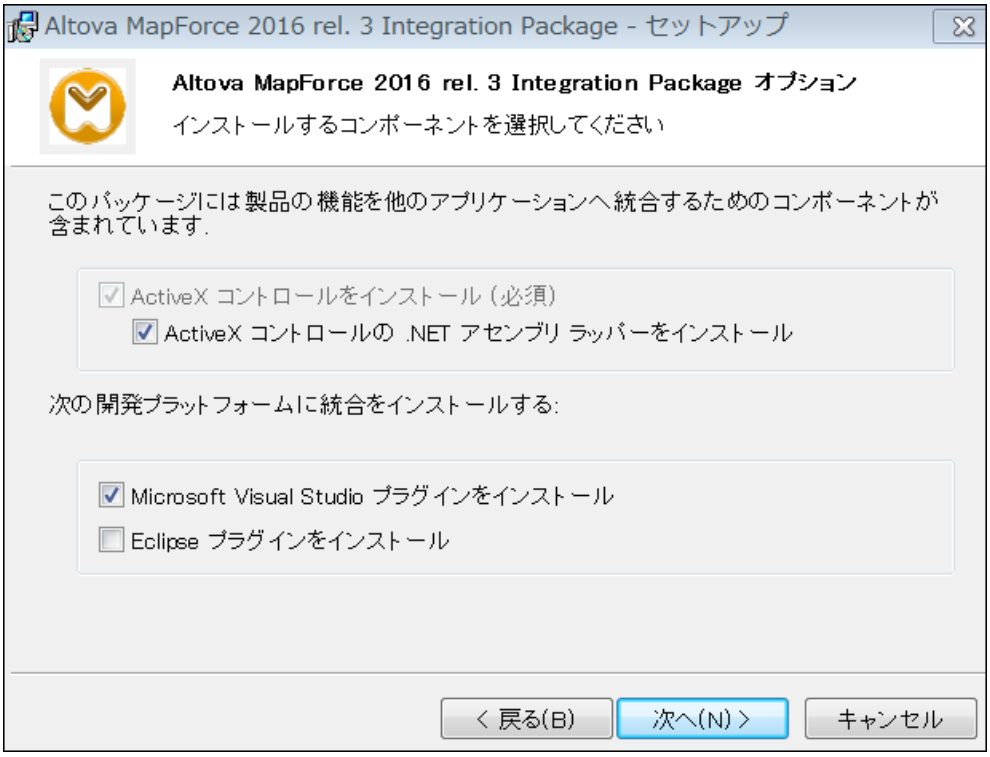
13.1 プラグインの有効化

必要条件：

- Microsoft Visual Studio 2005/2008/2010/2012/2013/2015/2017
- MapForce (Enterprise または Professional Edition)
- MapForce Integration Package、以下でダウンロードすることができます http://www.altova.com/ja/download/mapforce.html。

メモ： MapForce を Visual Studio プラグインとして使用するには、32 ビットの MapForce と MapForce Integration Package をインストールする必要があります。Visual Studio 内での 64 ビットプラグインはサポートされません。

Visual Studio のための MapForce プラグインを有効化するには、MapForce Integration Package をダウンロードして、実行し、スクリーンのインストールの方法に従ってください。インストール中、**「Microsoft Visual Studio をインストール」プラグイン**オプションが選択されていることを確認してください：



プロンプトされると、プラグインが有効化されるべき Visual Studio バージョンを選択します。例：



メモ: 使用中のオペレーティングシステムにインストールされているVisual Studio のバージョンのみが選択のために表示されます。

Integration Package がインストールされると、MapForce 機能 がVisual Studio 環境内で使用することができるようになります。

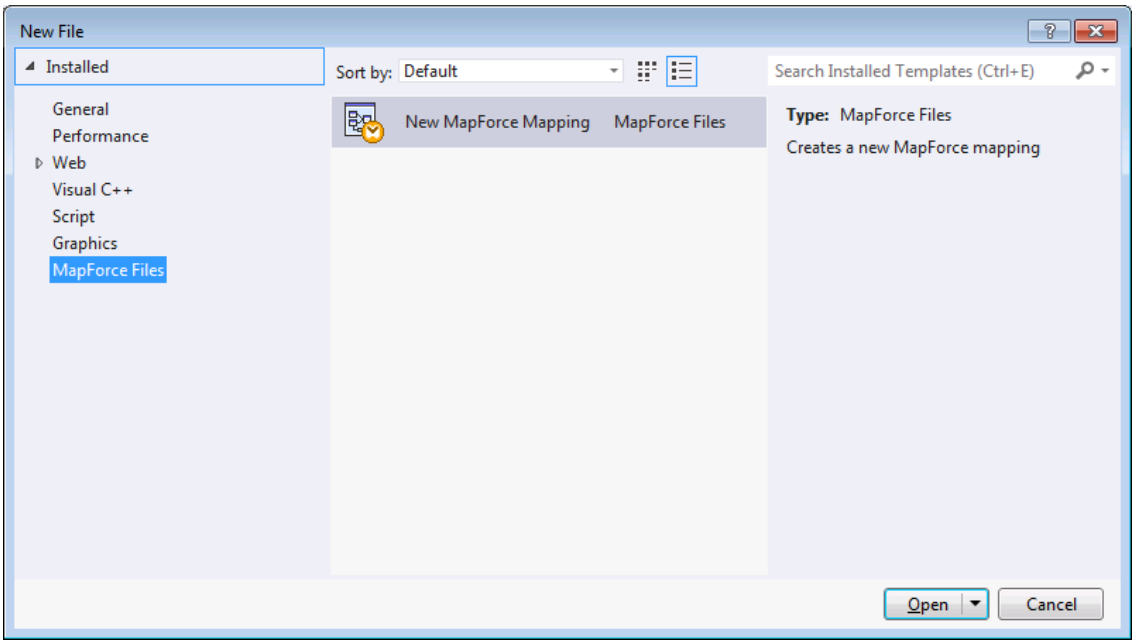
MapForce プラグインを手動で有効化する

インストール中にプラグインが自動的に有効化されない場合があります。有効化するには、以下を行います :

1. Visual Studio IDE 実行可能ファイルがインストールされているディレクトリまでナビゲートします、 例 :c:\Program Files\Microsoft Visual Studio 8\Common7\IDE)。
2. 管理者としてコマンドプロンプトを実行し、`devenv.exe /setup` を入力します。
3. Visual Studio 内のアプリケーションの使用を開始する前に、プロセスが通常に終了するまで待ってください。

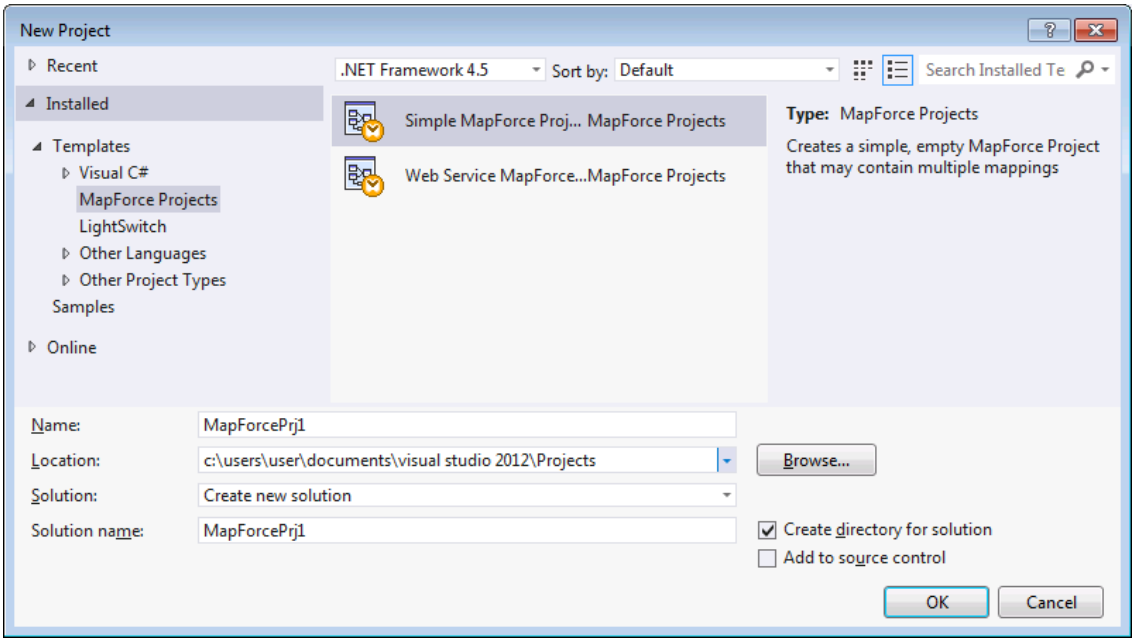
13.2 プラグインの有効化

Visual Studio のための MapForce プラグインが有効化されている場合、スタンドアロン MapForce グラフィカルユーザーインターフェイスとは異なり、Visual Studio 環境に適用できる方法でマッピングとマッピングプロジェクトを作成、または、開くことができます。例：(**「ファイル | 新規作成」**メニューコマンドを使用して) Visual Studio 内に新規のファイルを作成すると、または (**「プロジェクト | 新しいアイテムを追加」**メニューコマンドを使用して新規アイテムをプロジェクトに追加すると、**MapForce ファイル**型として選択することができます。



新規ファイル ダイアログボックス (MapForce Enterprise エディションプラグインを使用した Visual Studio 2012)

同じような方法で、新規の Visual Studio プロジェクトを作成すると、**「MapForce プロジェクト」**をプロジェクトテンプレートとして選択することができます。以下のスクリーンショットは、MapForce Enterprise Edition プラグインが有効化されている Visual Studio 2012 内のサンプル新規プロジェクトダイアログボックスを表示しています。



新規プロジェクトダイアログボックス (MapForce Enterprise エディションプラグインを使用したVisual Studio 2012)

既存のMapForce ファイルとプロジェクトは、Visual Studio ネイティブ機能からも開くことができます。既存の マッピングファイル または プロジェクトを開く場合、適用することのできるVisual Studio メニューを使用します (例：**「ファイル| 開く| ファイル」** または **「ファイル| 開く| プロジェクト/ソリューション」**) を使用して、 MapForce-に関連した ファイルの型を検索します。

13.3 共通メニューと関数へのアクセス

Visual Studio のための MapForce プラグインが有効化されている場合、下で表示されるように共通メニューと関数にアクセスすることができます。これはデフォルトのセットアップですが、Visual Studio の **「ツール | オプション」** メニューからメニューの場所を変更することができます。

グローバルリソース

MapForce グローバルリソースは、Visual Studio 内の **「MapForce | グローバルリソース」** メニューで使用できます。

MapForce オプション

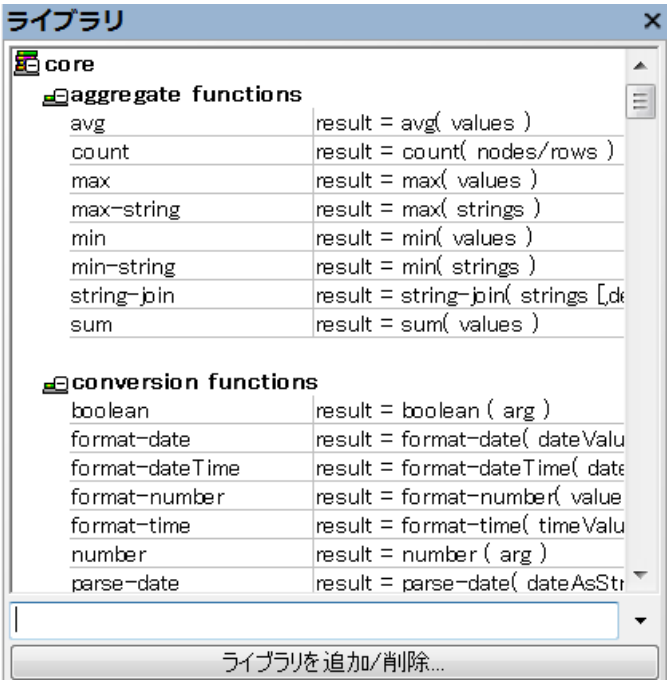
MapForce オプションは、Visual Studio 内の **「ツール | MapForce オプション」** メニューで使用できます。

マッピングペインのカスタマイズ

Visual Studio のメインペインで MapForce マッピングが開かれると、**「表示 | MapForce」** メニューを使用することができます。MapForce のスタンドアロンバージョンと同じオプションが含まれます。

ライブラリウィンドウ

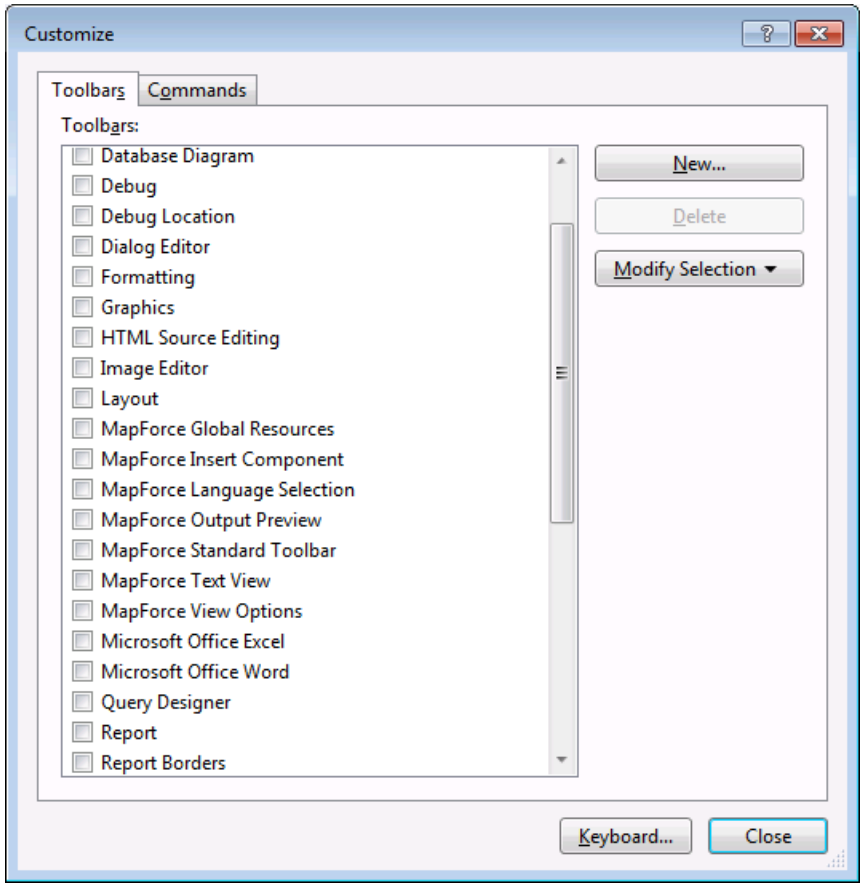
MapForce ライブラリウィンドウは、プラグインがインストールされた後、デフォルトでは Visual Studio 内で有効化されていません。このウィンドウで頻繁に作業する場合、か **「表示 | MapForce | ライブラリウィンドウ」** メニューから有効化することができます（このメニューは、メインウィンドウ内でマッピングファイルが開かれると、Visual Studio 内で使用することができます）。ライブラリウィンドウが有効化されると、インターフェイスの特定の箇所に、他の Visual Studio のコンポーネントと同様ドッキングすることができます。



ライブラリウィンドウ (Visual Studio 2012 with MapForce Enterprise エディション プラグイン)

ツールバーとコマンドのカスタマイズ

MapForce メニューとツールバーを Visual Studio の **「ツール | オプション」** メニューからカスタム化することができます。 .



ダイアログボックスのカスタマイズ (MapForce Enterprise Edition プラグインを使用したVisual Studio 2012)

ヘルプとサポート

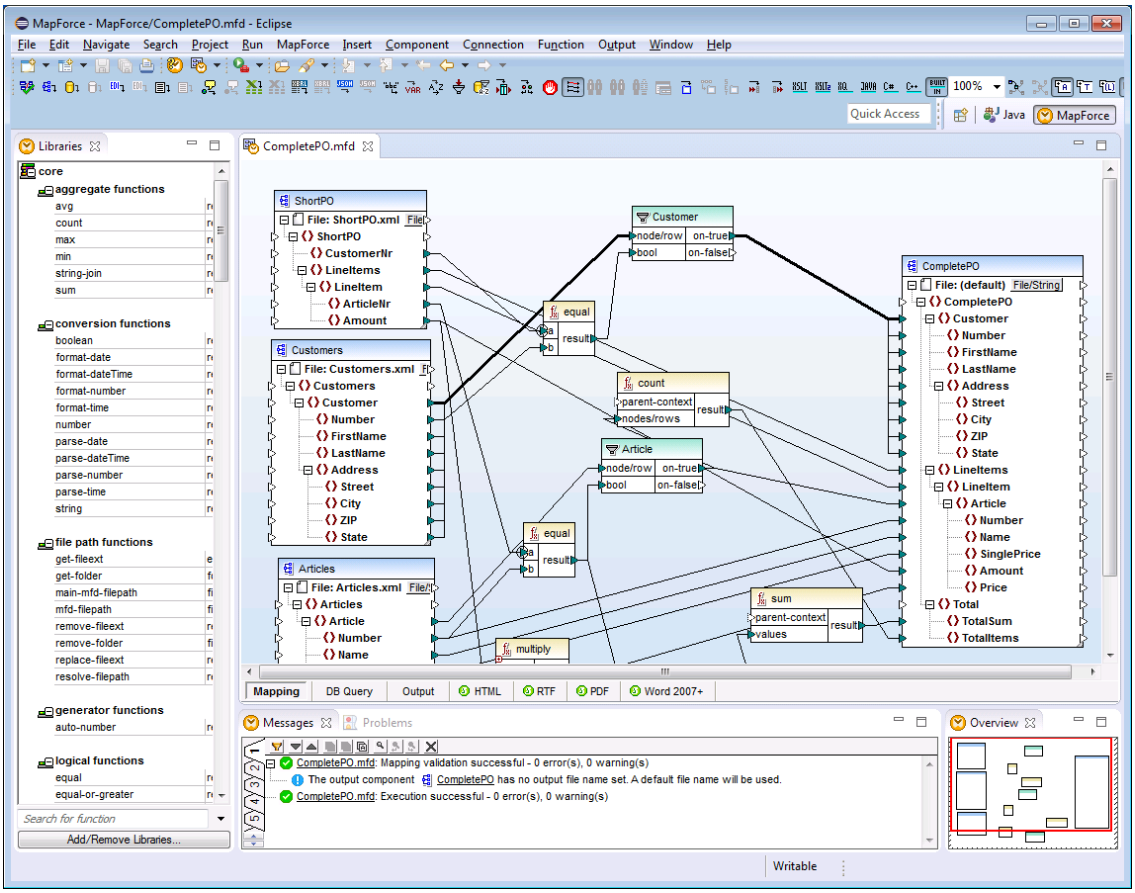
Visual Studio の **「ヘルプ| MapForce ヘルプ」** メニューでMapForce ヘルプ サポートセンター、最新情報のチェック およびMapForce についてのメニューを使用することができます。

Chapter 14

Eclipse のための MapForce プラグイン

14 Eclipse のための MapForce プラグイン

Eclipse は、プラグインのフォーム内で届けられる異なる種類のアプリケーションを統合するオープンソースのフレームワークです。MapForce Enterprise とProfessional Edition をEclipseバージョン 4.4 /4.5 /4.6に統合し、Eclipse から直接 MapForce 機能にアクセスすることができます。



MapForce Enterprise Edition プラグイン (Eclipse 4.4)

以下のトピックは、はのインストールおよび使用について説明しています。

- Eclipse のためのMapForce プラグインのインストール
- MapForce パースペクティブ
- 共通メニューと関数へのアクセス
- マッピングとプロジェクトとの作業
- Eclipse のためのMapForce プラグインの有効化

14.1 Eclipse のための MapForce プラグインのインストール

Eの必要条件：

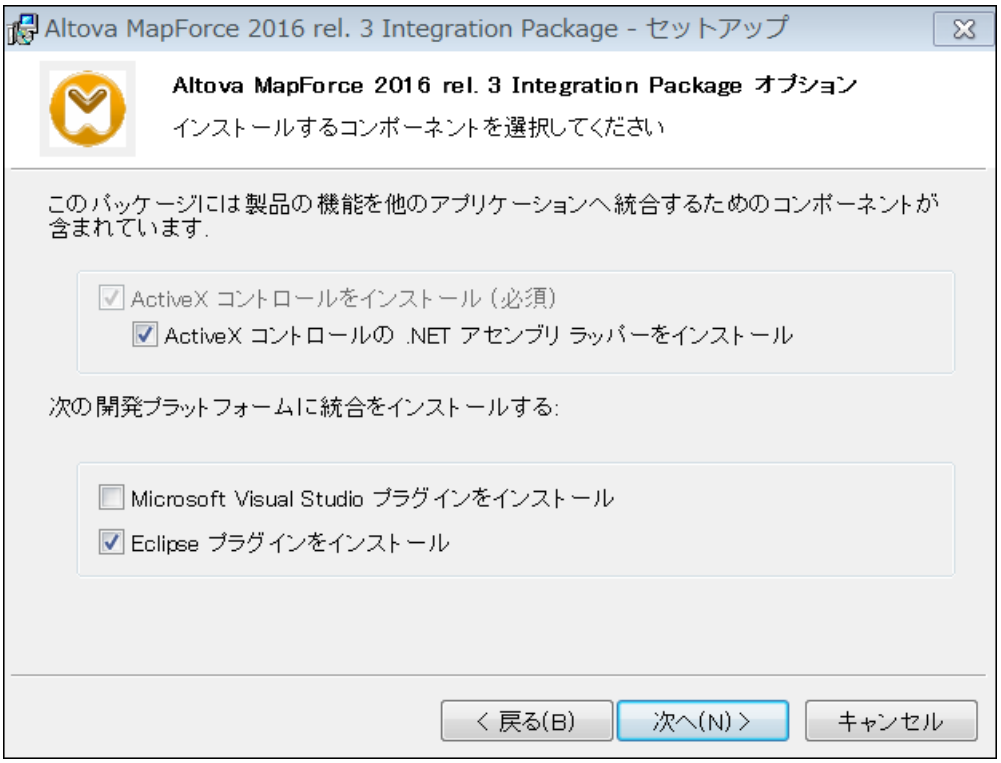
- Java Runtime Environment (JRE) 6.0 または以降（http://www.oracle.com/technetwork/java/javase/downloads/index.htmlを参照してください）。MapForce (32-ビット または 64-ビット)のバージョンに一致する 32-ビット または 64-ビット JRE をインストールしてください。
- Eclipse プラットフォーム4.4 / 4.5 / 4.6 (http://www.eclipse.orgを参照してください)。MapForce (32-ビット または 64-ビット)のバージョンに一致する 32-ビット または 64-ビット Eclipse をインストールしてください。
- MapForce Enterprise または Professional Edition のインストール。

Eclipse のための MapForce プラグインのインストール

1. MapForce Integration Package からダウンロードページ（http://www.altova.com/ja/download_components.html ）をダウンロードします。
2. Eclipse が実行されていないことを確認して、ダウンロードパッケージを実行します。

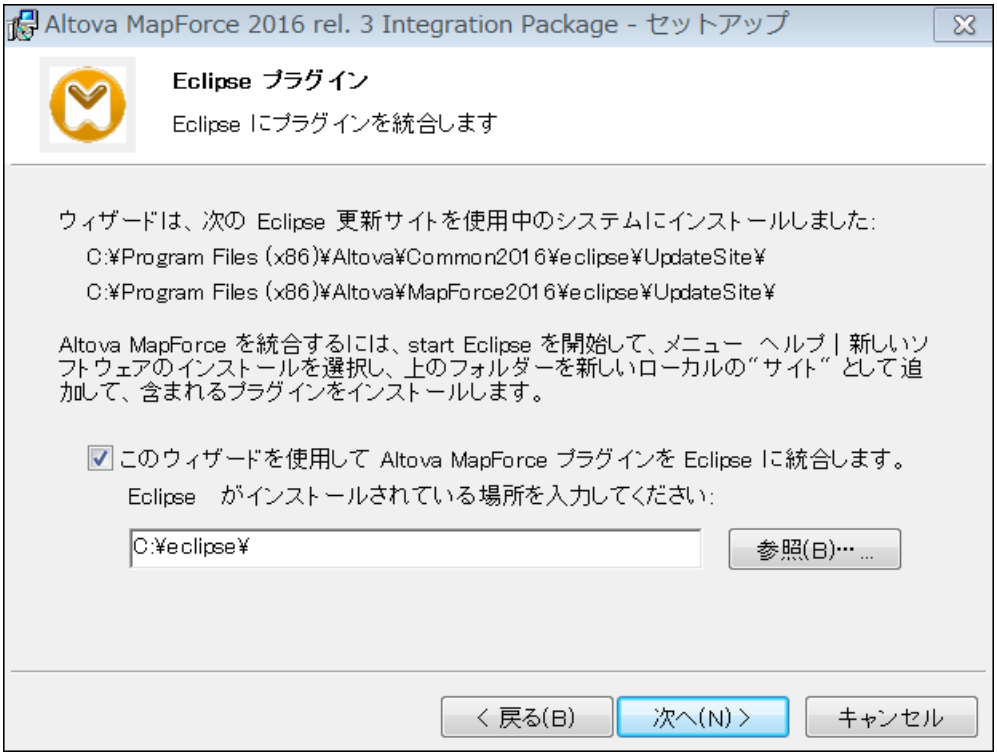
> MapForce Integration Packageのインストール、または、アンインストール中、Eclipse は閉じられている必要があります。

3. プロンプトされると、 **Eclipse プラグイン** オプションのインストールを選択し、 **『次へ』** をクリックします。



4. MapForce プラグインのインストール方法を問われると、 Eclipse に統合を選択し、以下を行います：
   a. プラグインのインストールを自動的に完了するには、このオプションを選択します（これは奨励されるオプションです）。 **このウィザードを使用して、Altova MapForce プラグインを Eclipse に統合する** を選択し、Eclipse 実行可能ファイル(`eclipse.exe`) が存在するディレクトリを参照します。

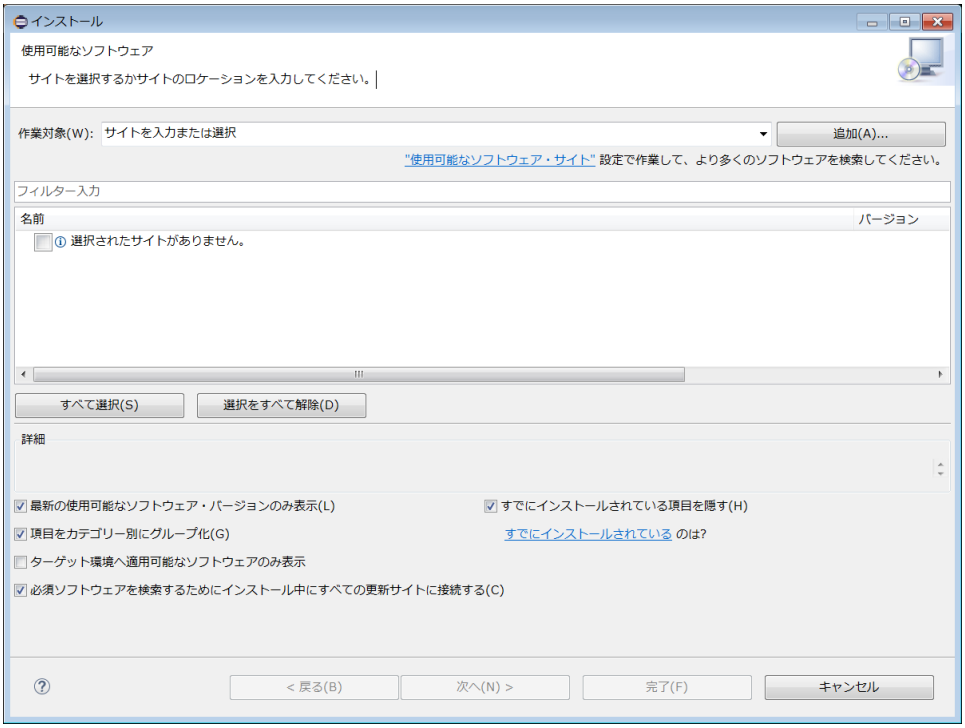
b. Eclipse 内で個別にプラグインのインストールを行うには、**このウィザードを使用して、Altova MapForce プラグインを Eclipse に統合する**のチェックボックスを解除します。をチェックします。 (下の説明を参照)。



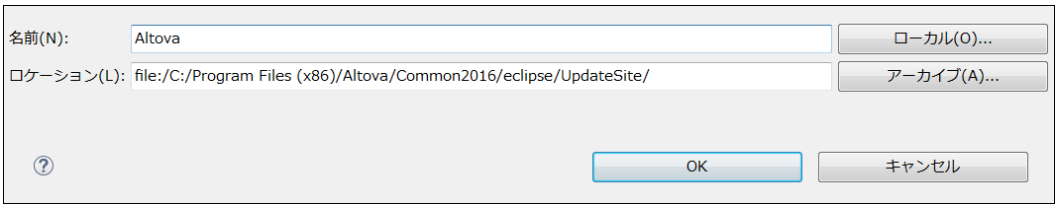
2. **次へ**をクリックして、インストールを完了します。自動的な統合を選択した場合、 MapForce パースペクティブとメニューは、次回 Eclipse が開始されると、表示されます。

Eclipse のための MapForce プラグインを手動で統合する

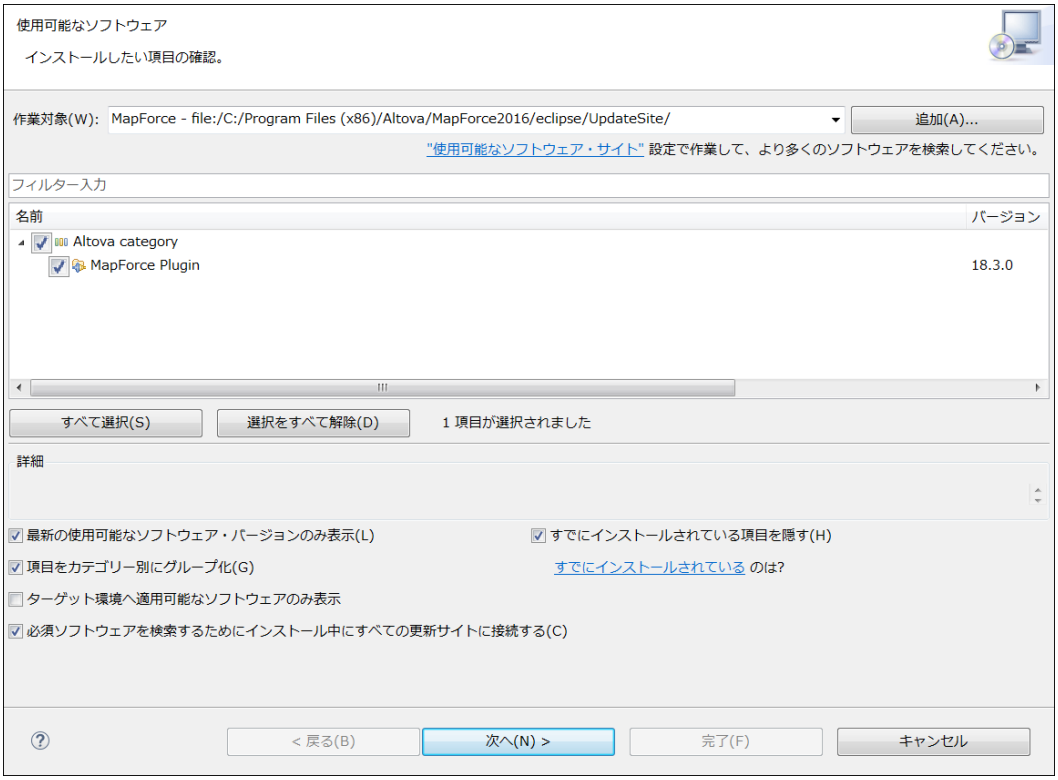
1. Eclipse 内でメニューコマンド **ヘルプ | 新規ソフトウェアのインストール**をクリックします。
2. ポップアップされるインストール ダイアログ内で (下のスクリーンショット) **追加** ボタンをクリックします。



3. ポップアップされるリポジトリの追加ダイアログ (*下のスクリーンショット*) で **ローカル** ボタンクリックします。
4. フォルダー `c:\Program Files\Altova\Common2017\eclipse\UpdateSite` を参照します。('Altova' などの) サイトの名前を入力し、 **「OK」** をクリックします。



5. *ステップ 2 から 4 を繰り返します、今度は、フォルダー* `c:\Program Files\Altova\MapForce2017\eclipse\UpdateSite` *を選択し、 Altova MapForce' などの名前を与えます。*
6. *インストールダイアログの作業対象コンボボックスから、 -- ローカルサイトのみ --を選択します (下のスクリーンショット参照)。これにより、下に表示されるように使用することのできるすべてのプラグインが下のペインに表示されるようになります。Altova category フォルダーのトップレベルのチェックボックスのチェックボタンをチェックします。(下のスクリーンショット参照)。* **「次へ」** *ボタンをクリックします。*

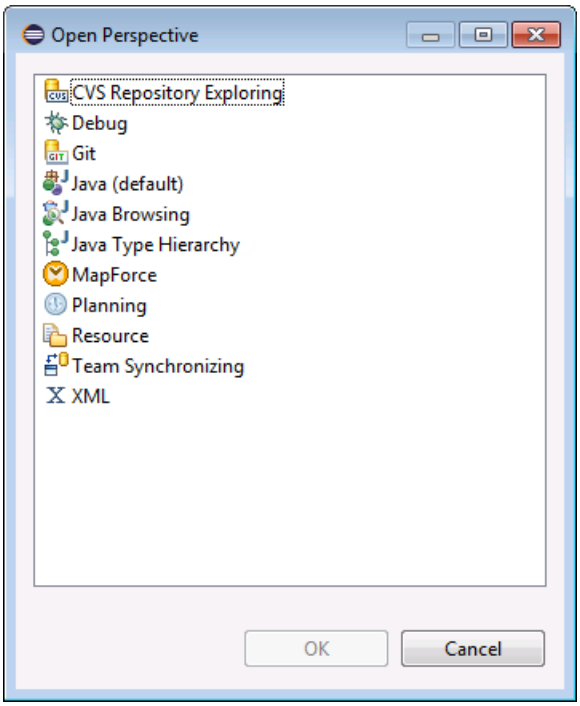


7. *インストールの詳細* スクリーンによりインストールされるアイテムをレビューすることができます。 **「次へ」**を押して、続行します。
8. *ライセンスのレビュー* スクリーンが表示されます。使用許諾契約書に同意するを選択します（MapForce プラグインのためにMapForce Enterprise または Professional Edition ライセンス以外の追加ライセンスは必要ありません）。 **「完了」**をクリックしてインストールを完了します。

メモ: プラグインで問題が生じた場合は、（例えば、不足しているアイコン）コマンドラインからクリーンなフラグを使用して Eclipse を起動してください。

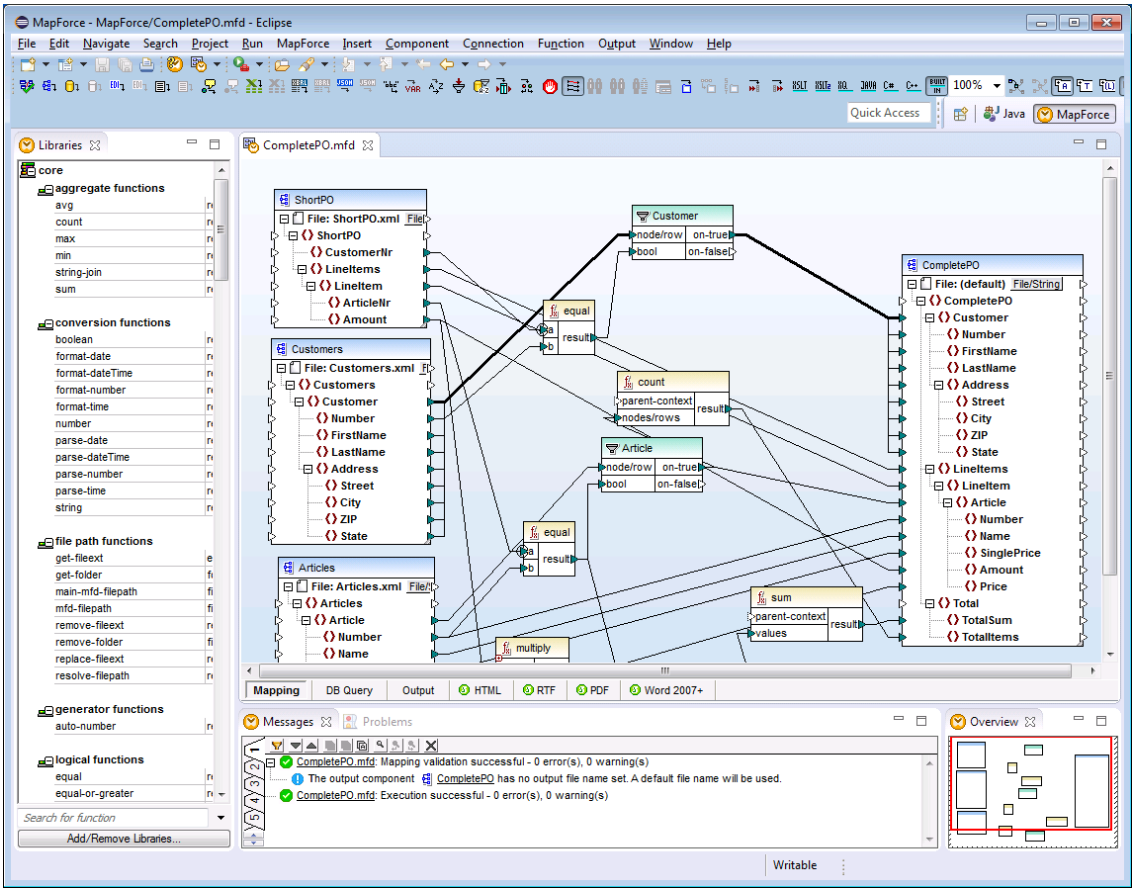
14.2 MapForce パースペクティブ

Eclipse のためのMapForce プラグインをインストールすると、Eclipse 内で新しいパースペクティブ ("MapForce")を使用することができるようになります。このパースペクティブのレイアウトは、MapForce のスタンドアロンエディションのインターフェイスに類似しています。MapForce パースペクティブに切り替えるには、**「ウィンドウ| パースペクティブを開く| その他」**をクリックし、MapForce をリストから選択します。



Eclipse 内でMapForce パースペクティブを選択する

MapForce パースペクティブは、Eclipse の他のパースペクティブと同じで、必要に応じてEclipse 内 **(ウィンドウ| ナビゲーション| 次へ パースペクティブ)** で切り替えます。含まれるビューをカスタム化することもでき、Eclipse 設定から他の複数のオプションをカスタム化します。 Eclipse 4.4 内でMapForce パースペクティブをカスタム化するには、MapForce パースペクティブに切り替え、メニューコマンド **「ウィンドウ| パースペクティブのカスタム化」**)を選択します。 Eclipse パースペクティブに関しての詳しい情報は、Eclipse ドキュメントを参照してください。以下のスクリーンショット MapForce パースペクティブのスイッチがオンされているEclipse 環境を示しています。

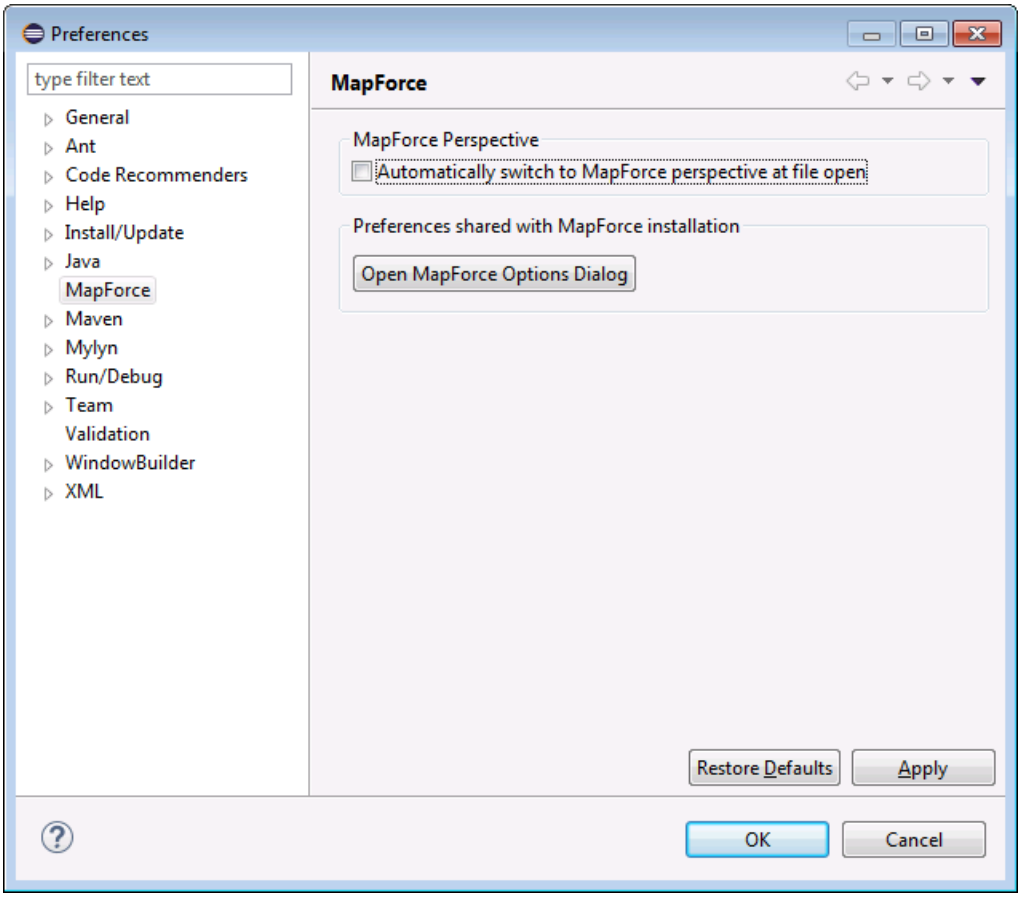


MapForce パースペクティブ (Eclipse のためのMapForce Enterprise Edition プラグイン)

デフォルトでは、Eclipse 内でのMapForce パースペクティブは以下のように整理されています：

- Eclipse エディターでマッピングデザインウィンドウを使用することができます。MapForcein のスタンドアロンエディション内と同じタブと機能が搭載されています。
- Eclipse ビューとして、メインマッピングエディター左のライブラリウィンドウを使用することができます。このビューが表示されていない場合、MapForce パースペクティブに切り替えます。メニューコマンド **『ウィンドウ| ビューの表示 | ライブラ』**を選択します。ライブラリビューにより、定義済みまたは カスタム定義された関数と関数ライブラリと作業できます。
- Eclipse ビューとして、メインマッピングエディター下のメッセージペインを使用することができます。メッセージビューが表示されていない場合、MapForce パースペクティブに切り替え、メニューコマンド **『ウィンドウ| ビューの表示 | メッセージ』** を選択します。メッセージは、検証メッセージ、エラー、と警告を表示します。
- Eclipse ビューとして、概要ペインを使用することができます。アウトラインビューが表示されていない場合は、MapForce パースペクティブに切り替え メニューコマンド **『ウィンドウ| ビューの表示 | アウトライン』** を選択します。このビューを使用することにより、大きなマッピングのマッピングデザインの特定の箇所に素早く移動することができます。

MapForce マッピングを開くと、Eclipse がMapForce パースペクティブに自動的に切り替えるように構成することもできます。これを行うには、メニューコマンド **『ウィンドウ| 設定』** を開き、MapForce を選択して、 **『ファイルが開かれると自動的にMapForce パースペクティブに切り替える』** チェックボックスを選択します。



設定 ダイアログボックス

14.3 共通メニューと関数へのアクセス

Eclipse 内では、下にリストされるEclipse 固有のバリエーションを除き、スタンドアロンバージョン内と同様にMapForce 機能の同じメニューの多数にアクセスすることができます。これはデフォルトのセットアップですが、 Eclipse からのインターフェイスの基本設定を更にカスタマイズする場合は、 (MapForce パースペクティブを参照してください)。

メモ: コンテキストが関連しない場合、Eclipse 内では、 MapForce メニューグループまたはコマンドの一部が無効化 (または使用できない)場合があります。例 :Eclipse 内でマッピングデザインファイル (mfd) がアクティブな場合のみ、 **挿入**」メニューを使用することができます。

MapForce 標準メニューに関する詳細は、 メニューレファレンスを参照してください。

一般的な MapForce コマンド

MapForcein のスタンドアロンエディション内で、マッピングデザインファイルに適用することのできる(**検証**」、 **「FlowForce Server にデプロイ**」、 **「コードの生成**」などの)コマンドは、 **「ファイル**」メニュー内で使用することができます。Eclipse 内でこれらのコマンドは**MapForce** メニュー内またはMapForce ツールバーで使用することができます。(MapForce プロジェクトファイルを含む)ファイルを開いたり閉じたりするコマンドは、Eclipse の **「ファイル**」メニューで使用することができることに注意してください。



Eclipse 内のMapForce ツールバー

ツールバーボタンは、MapForce ヘルプファイルを開きます。

ツールバーボタンは、 MapForce ファイル固有のコマンドを表示します。このボタンを展開すると、現在 Eclipse エディター上でアクティブなファイルにより決定される使用可能なコマンドが表示されます。例 :マッピングデザイン (mfd) ファイル固有のコマンドは、このようなファイルがでアクティブな場合、Eclipse エディター内で使用することができます。

グローバルリソース

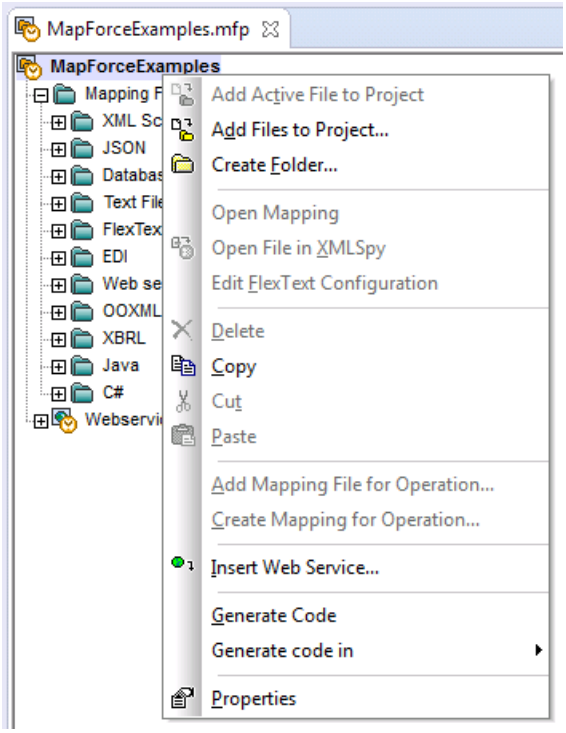
グローバルリソースへアクセスまたはグローバルリソースの管理は、以下を行います :

- MapForce ツールバーボタンをクリックして展開し、**グローバルリソース**をクリックします。
- MapForce メニューから**グローバルリソース**をクリックします。

MapForce プロジェクト

MapForce のスタンドアロンのエディションでは、**プロジェクト** メニューは、マッピング プロジェクト (mfp) ファイルに適用することのできる複数のコマンドを含んでいます。Eclipse 内では、これらのコマンドは以下のように存在します :

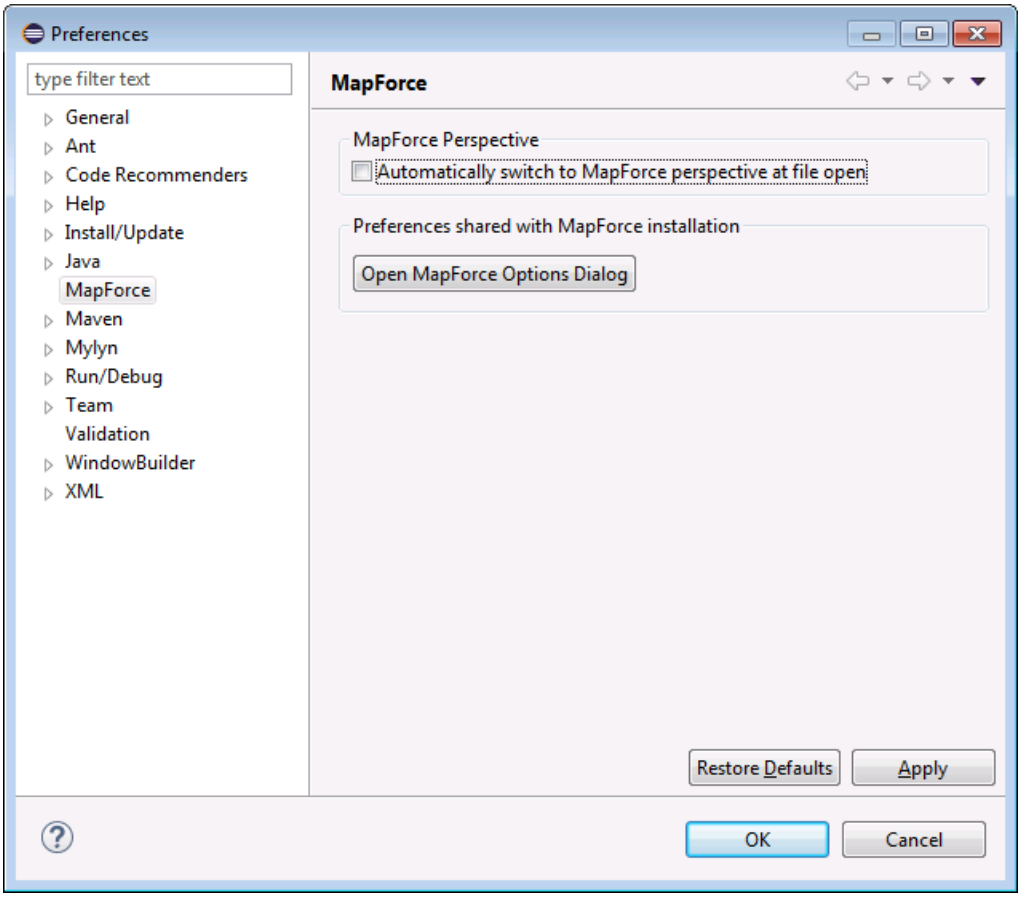
- プロジェクトを開くまたは閉じるコマンドは、 Eclipse **「ファイル**」メニューから使用することができます。
- 他のプロジェクト専用のコマンドは、Eclipse 内でコンテキストコマンドとして使用できます。コンテキストコマンドを表示、MapForce プロジェクト (mfp) ファイルを作成または開く場合は、プロジェクトを右クリックしてください。



標準 MapForce プロジェクト (mfp) に加え、Eclipse 内では、"MapForce/Eclipse" の型のプロジェクトも作成することができます。このようなプロジェクトは、2面の特質のため、MapForce コードの自動的なビルドと生成のために構成することができます。マッピングおよびプロジェクトと作業を参照してください.

MapForce オプション

MapForce オプションは、**「ウィンドウ| 優先」**メニューから使用することができます。優先ダイアログボックスで、**MapForce** を選択し、**「MapForce オプションダイアログを開く」**をクリックします。



設定ダイアログボックス

ライブラリウィンドウ

Eclipse 内で、MapForce ライブラリウィンドウは、ビューとして使用することができます。このビューはデフォルトでは、メインエディターウィンドウの左に配置されています。 (MapForce に関連したすべてのビューは、MapForce パースペクティブがオンに切り替えられると、Eclipse インターフェイスで表示されます MapForce パースペクティブも参照)。

MapForce プラグイン バージョン

Eclipse のための MapForce プラグインの現在インストールされているバージョンを確認するには、Eclipse メニューオプション **「ヘルプ | Eclipse プラットフォームについて」** を選択して、そして、MapForce アイコンを選択します。

ヘルプとサポート

MapForce ヘルプ、サポートセンター、最新情報 メニューは、Eclipse の **「ヘルプ| MapForce ヘルプ」** メニューで使用することができます。

14.4 マッピングおよびプロジェクトと作業

When Eclipse のためのMapForce プラグインがインストールされている場合、Eclipse のEclipse プロジェクトからMapForce のスタンドアロンエディション内と同じマッピング とマッピングプロジェクト型 を作成することができます。マッピングをデザイン、コンパイル、デプロイし、マッピングコードを生成するには、新規のEclipse プロジェクトを作成するか、または既存のEclipse プロジェクトを使用することができます (例 :MapForce マッピングを追加するJava プロジェクト)。

更に、 **MapForce/Eclipse プロジェクト**内の MapForce プラグインをインストール後にEclipse 内で使用することのできる、特別なプロジェクト型内で全てのマッピングと作業することができます。カスタマイズを選択しない限り、MapForce/Eclipse プロジェクトはデフォルトでJava ビルダーとMapForce コード生成ビルダーが割り当てられています。更に、2つのEclipse 性質が存在します :MapForce 性質と JDT (Java 開発ツール)性質。この結果、MapForce/Eclipse プロジェクトは、(マッピングデザインファイルなどの)リソースを保存または変更する場合、以下に記載されるように振る舞います :

- **プロジェクト > 自動的にビルド**」メニューオプションが有効化されていると、マッピングコードは自動的に生成されます。1つまたは複数のMapForce プロジェクトファイルMapForce/Eclipse プロジェクト内に存在する場合、コード生成言語と出力ターゲットフォルダーは、各プロジェクトファイル内の設定により決定されます。 それ以外の場合、Eclipse は、場所を選択するようにプロンプトします。
- エラーと出力メッセージは、メッセージと問題ビューに表示されます。

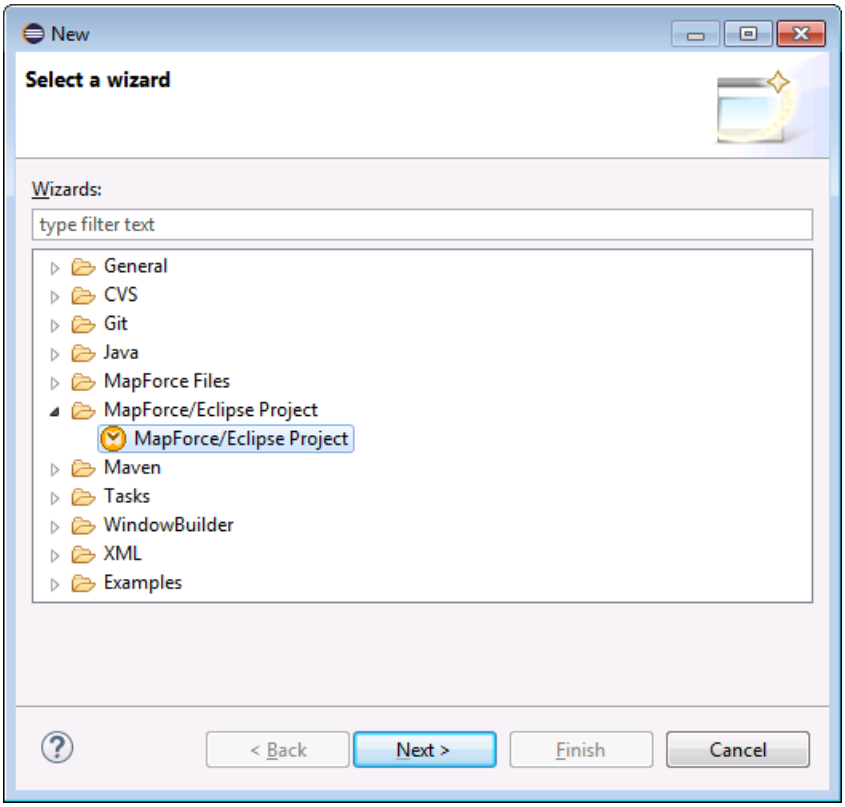
このセクションは、以下のトピックを含みます :

- MapForce/Eclipse プロジェクトの作成
- 新しいマッピングの作成
- Eclipse プロジェクトに既存のマッピングをインポートする
- 自動ビルドの構成とMapForce コードの生成

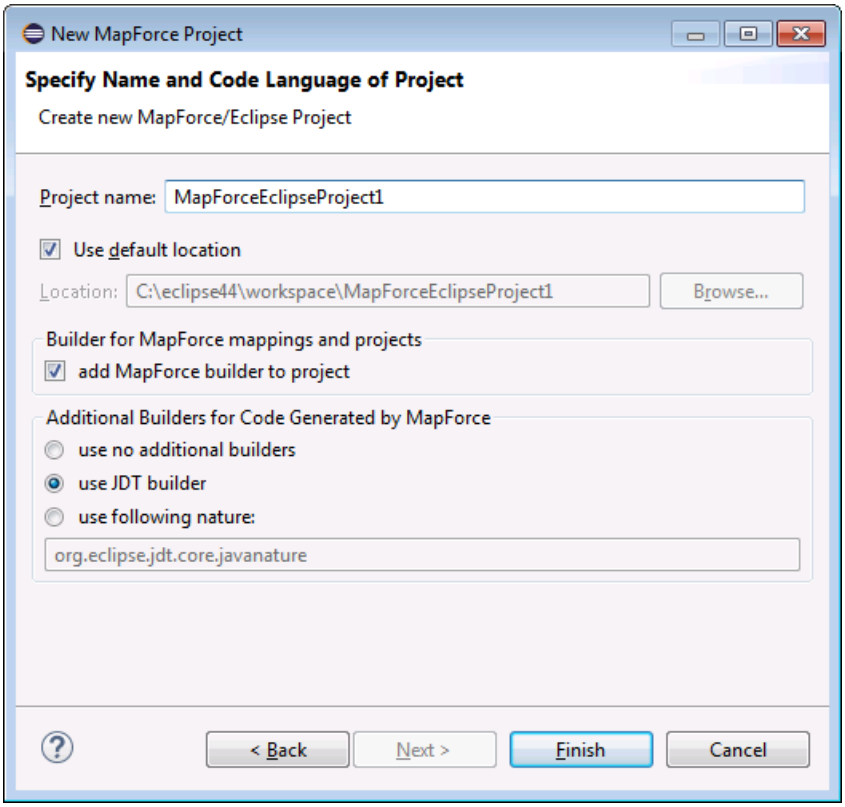
14.4.1 MapForce/ Eclipse プロジェクトの作成

MapForce/Eclipse プロジェクトの作成方法 :

1. **「ファイル」**メニューから **新規作成 | その他」**をクリックします。
2. **「MapForce/Eclipse プロジェクト」**カテゴリーを選択します。



3. **次へ** をクリックします。



4. プロジェクト名を入力して、プロジェクトの保存場所を選択します。 **MapForce ビルダーをプロジェクトへ追加** そのままにして、**JDT ビルダーを使用** オプションをそのままにします。
5. **「完了」**をクリックします。

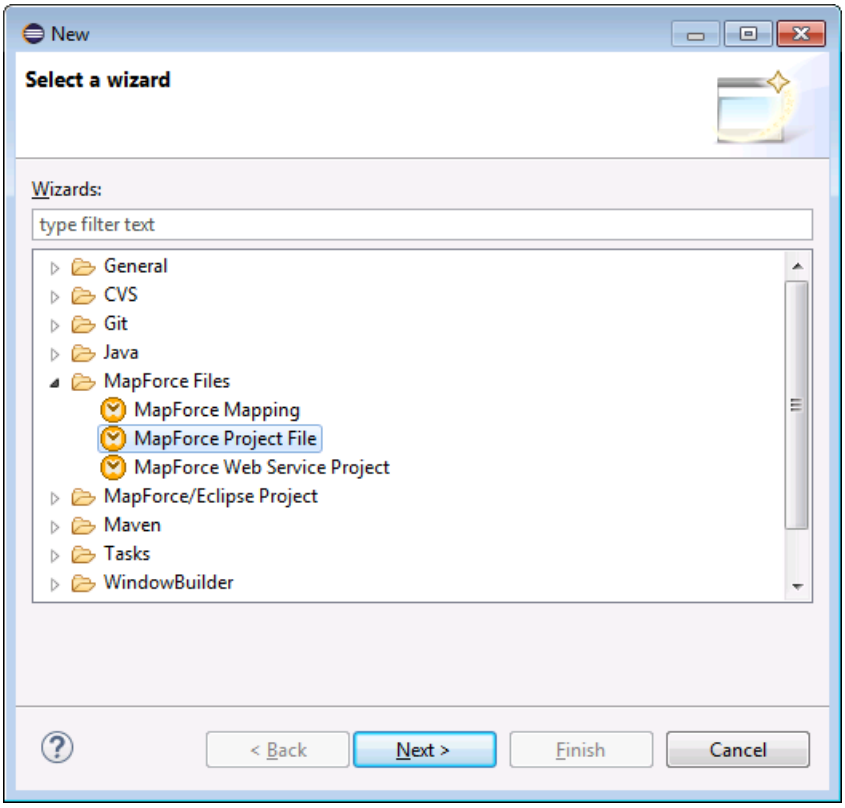
14.4.2 新しいマッピングの作成

以下のMapForce ファイル型をEclipse プロジェクト内で作成することができます：

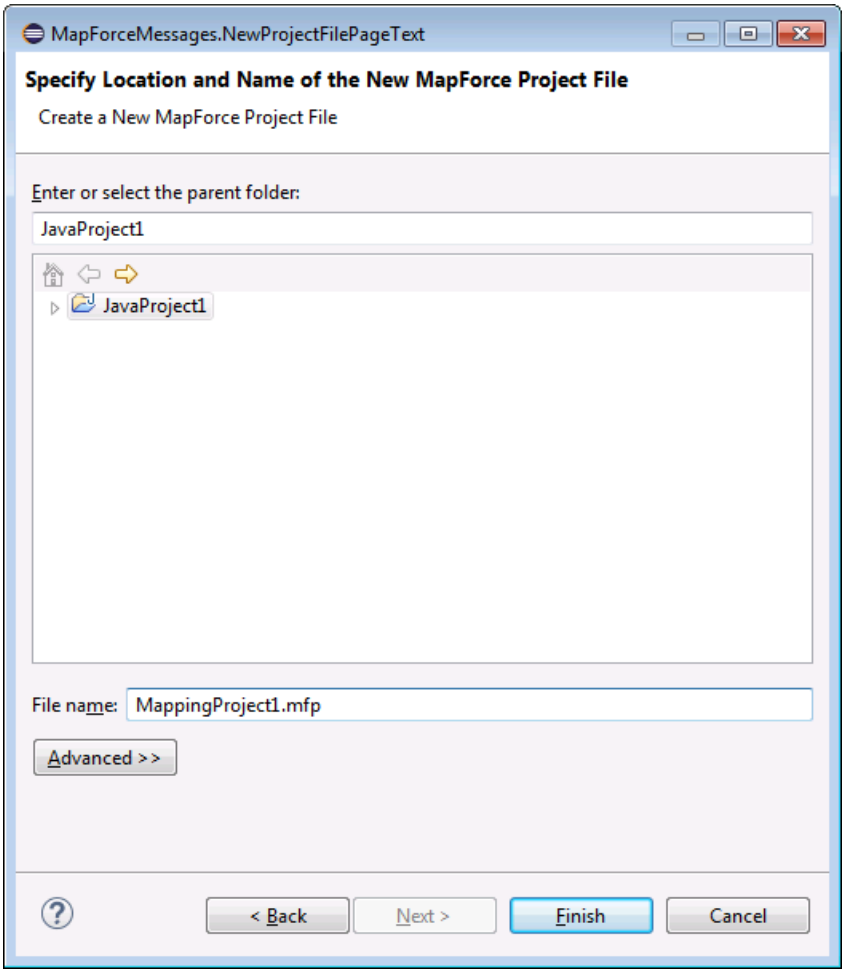
- MapForce マッピング
- MapForce プロジェクトファイル
- MapForce Web サービス プロジェクト (MapForce Enterprise Edition で使用することができます)

Eclipse プロジェクト内でこれらのファイル方を作成する：

1. 新規のEclipse プロジェクトを作成、または既存のプロジェクトを開きます。
2. **「ファイル」**メニューから **新規作成」**をクリックし、**「その他」**をクリックします。



3. ウィザードダイアログボックスから必要なファイル方を選択し、 **「次へ」**をクリックします。

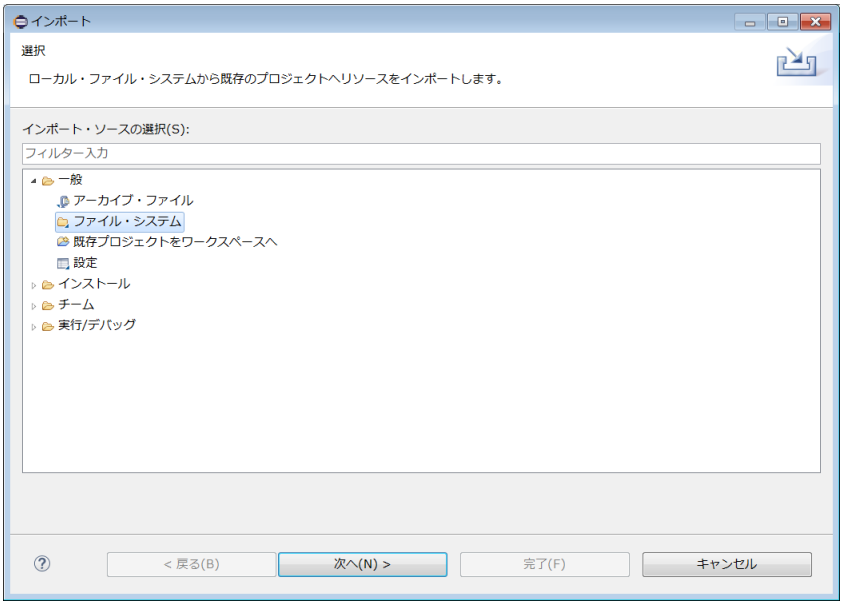


4. 既存のプロジェクト内の親フォルダーを選択して、**「完了」**をクリックします。

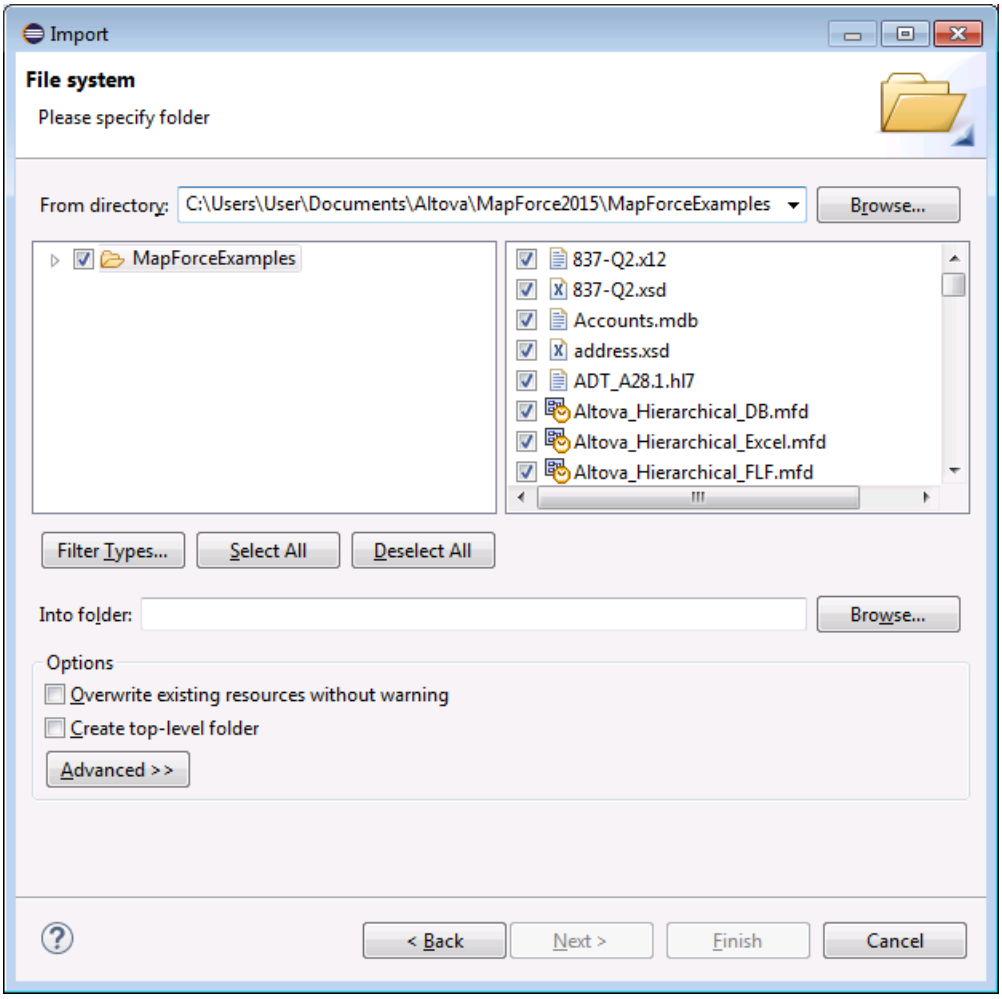
14.4.3 Eclipse プロジェクトに既存のマッピングをインポートする

MapForce マッピングと依存ファイルを既存のEclipse プロジェクトにインポートする:

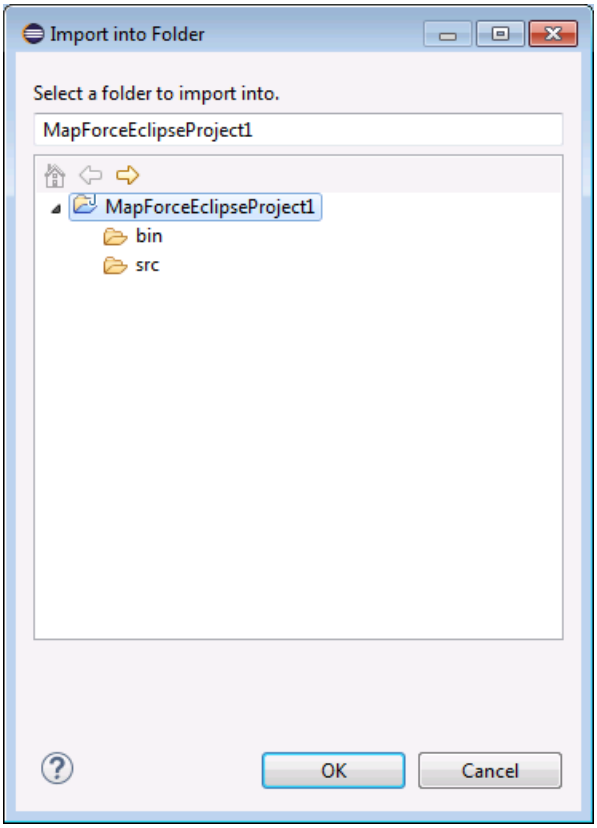
1. ファイルをインポートするプロジェクトを開きます。
2. **「ファイル」**メニューから**「インポート」**をクリックします。



3. **「ファイルシステム」**を選択して、**「次へ」**をクリックします。



4. **「次のディレクトリから」**の横で、インポートするファイルの場所を参照し、必要なファイルを選択します。
5. **「宛先フォルダー」**の横の、**「参照」**をクリックして、ファイルを追加するプロジェクトを選択します（このサンプルでは、*MapForceEclipse プロジェクト1*）。



6. **「OK」**をクリックして、**「完了」**をクリックします。

14.4.4 自動ビルドの構成と MapForce コードの生成

自動的な MapForce コードのビルドと生成は、MapForce/Eclipse プロジェクト内でデフォルトとして有効化されています（MapForce/Eclipse プロジェクトの作成 を参照してください）。*MapForce/Eclipse* 型ではない既存のプロジェクト内の MapForce コードの自動ビルドとコードの作成を行うには、*MapForce コード生成ビルダー*と*MapForce ネイチャー*を手動で追加して行います。

MapForce コード生成ビルダーをプロジェクトに追加する方法：

- Eclipse **.project** ファイル、以下でハイライトされているラインを追加します：

```
<buildSpec>
    <buildCommand>
       <name>org.eclipse.jdt.core.javabuilder</name>
       <arguments>
       </arguments>
    </buildCommand>
    <buildCommand>
       <name>com.altova.mapforceeclipseplugin.MapForceBuilder</name>
       <arguments>
       </arguments>
```

```
        </buildCommand>
    </buildSpec>
```

MapForce のネイチャーをプロジェクトに追加する：

- Eclipse **.project** ファイル、以下でハイライトされているラインを追加します：

```
    <natures>
        <nature>org.eclipse.jdt.core.javanature</nature>
        <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>

    </natures>
```

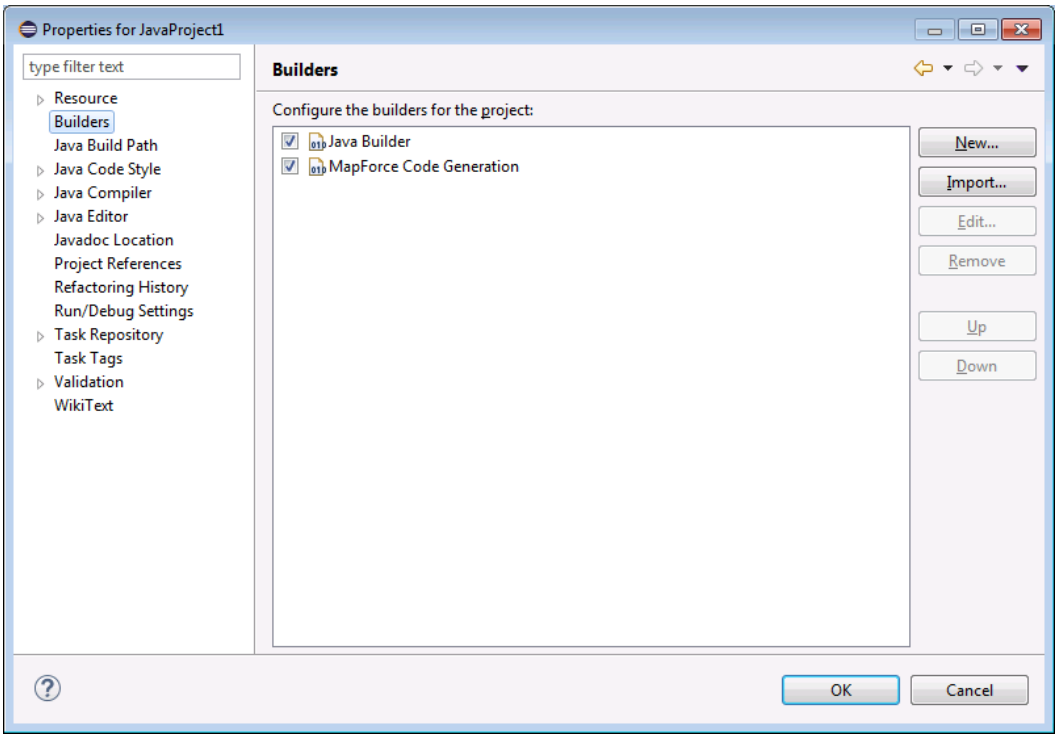
ヒント：.project ファイルを素早くEclipse のナビゲータービューから開くことができます（このビューを有効化するにはメニューコマンド **「ウィンドウ | ビューの表示 | ナビゲーター」** を選択します）。

自動的な MapForce コード生成のオン/オフの切り替え：

- **プロジェクト** メニューから**自動的にビルド**をクリックします。

MapForce コード生成ビルダーの無効化：

1. **プロジェクト** メニューから **「プロパティ」** をクリックします。
2. **「ビルダー」** をクリックします。



3. **「MapForce コード生成」** チェックボックスのチェックを解除するためにクリックします。

14.5 Eclipse のための MapForce プラグインの有効化

Eclipse のための MapForce プラグインは、ID を持つ Eclipse 拡張点 **com.altova.mapforceeclipseplugin** を提供します。この拡張点を使用して、MapForce プラグインの機能性を適用または拡張します。拡張点は、MapForce コントロールと MapForce API の COM-Interface へのアクセスを提供します。

MapForce Eclipse インストールパッケージは、プラグインの簡単なサンプルを含んでいます。新規 MapForce マッピングのファイルを開くイベントをチェックし、マッピングビューのズームのレベルを 70% に設定します。

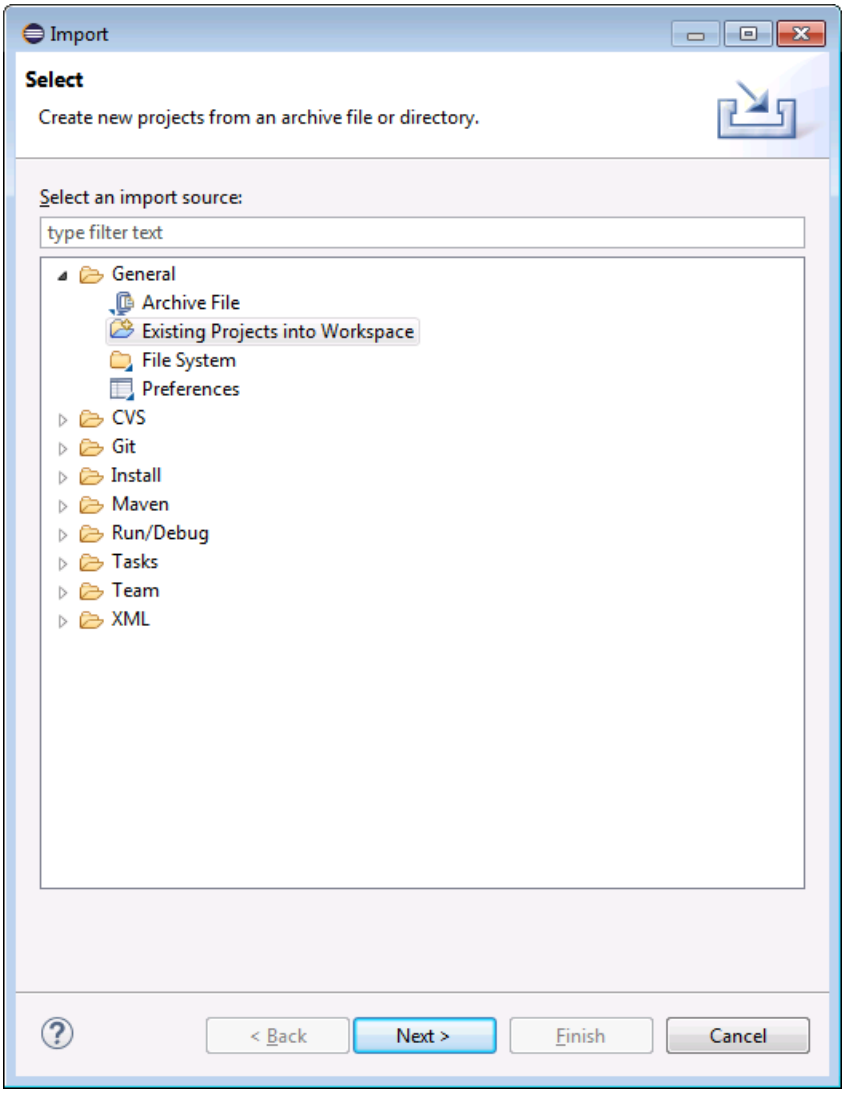
拡張ポイントの JavaDoc ドキュメントは、MapForce プラグイン インストールディレクトリ (通常、**C:\Program Files\Altova\MapForce2017\eclipse\docs**) で検索することができます。

サンプル MapForce プラグインをインストールして実行する前に、以下の必要条件が満たされていることを確認してください:

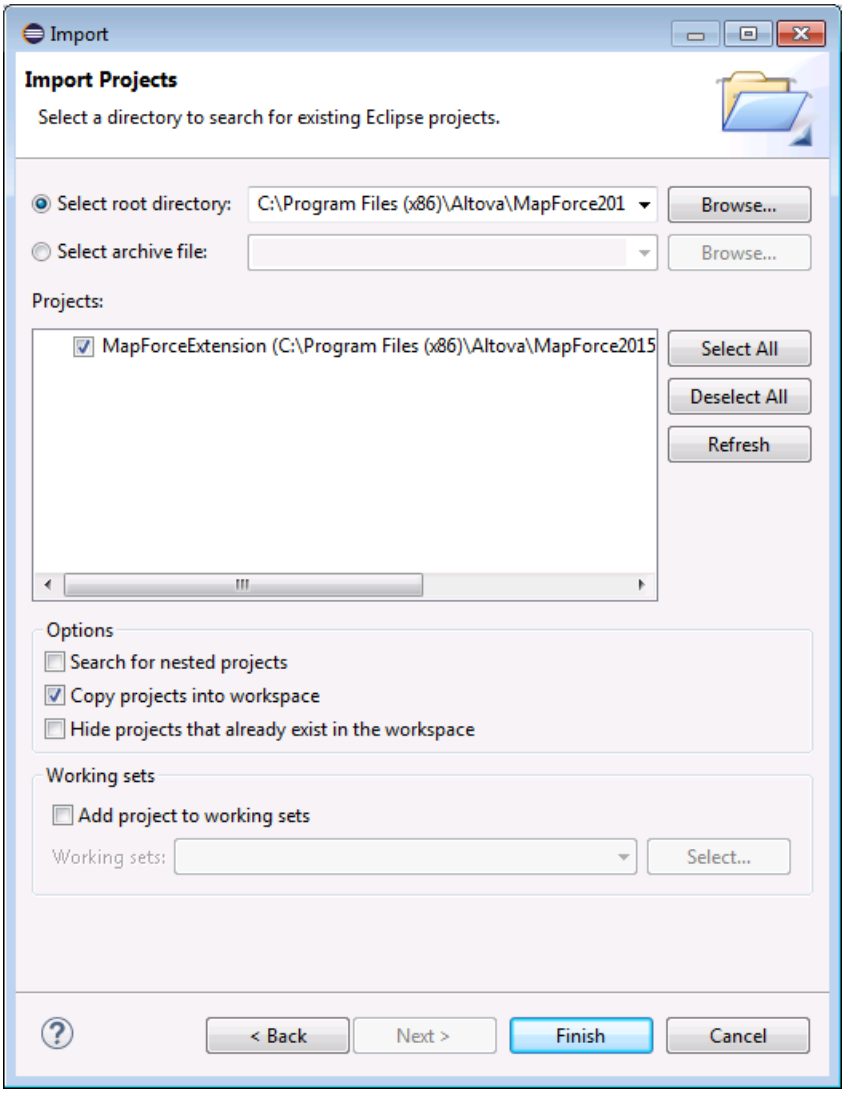
- 32-ビット Java を使用している場合は、32 ビット Eclipse、32 ビット MapForce および 32 ビット MapForce Integration Package が必要です。
- JDT (Java Development Tools) プラグインのインストール。
- Eclipse PDE (プラグイン開発環境) のインストール。

サンプル MapForce プラグイン プロジェクトをワークプレースにインポートする:

1. Eclipse を開始します。
2. **「ファイル」** メニューから **「インポート」** をクリックします。
3. **「一般 | 既存のプロジェクトをワークプレースへ」** を選択して、**「次へ」** をクリックします。



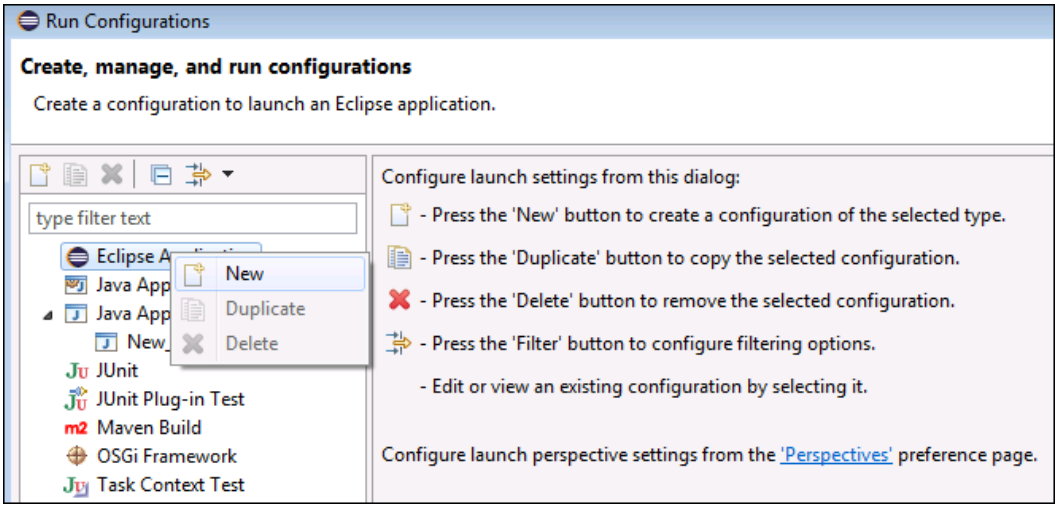
4. "ルートディレクトリの選択 "フィールドの横の **参照**... ボタンをクリックして、サンプルプロジェクトディレクトリを選択します。例 :**C:\Program Files\Altova\MapForce2017\eclipse\workspace\MapForceExtension.**



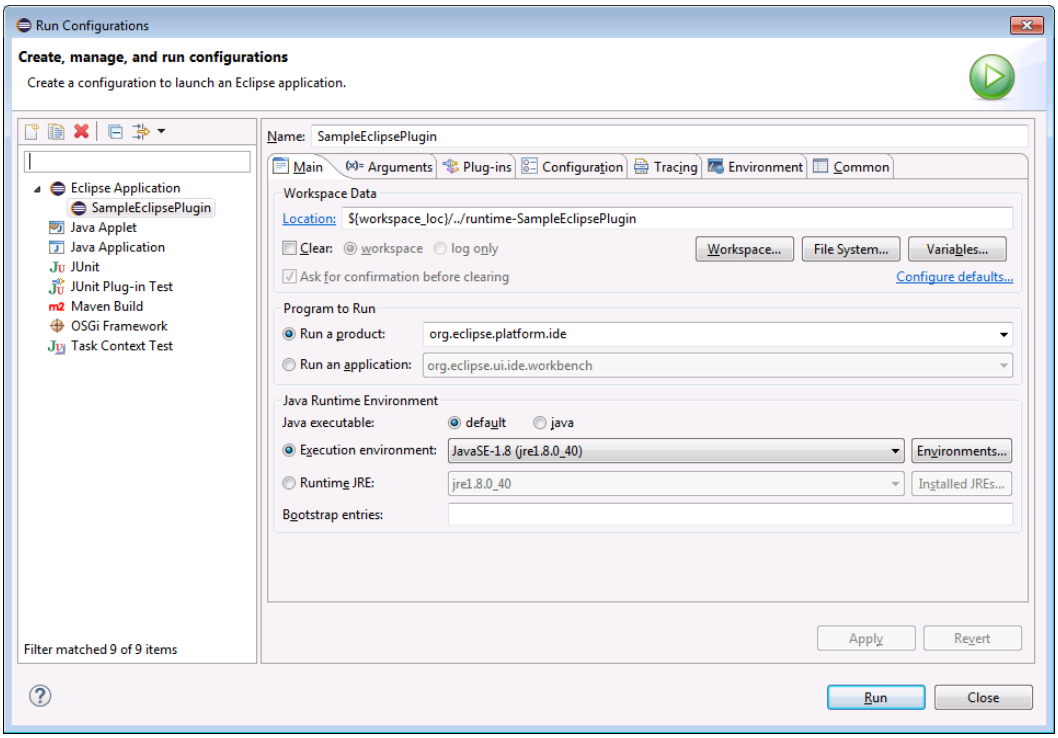
5. **「プロジェクトをワークプレースへコピー」**オプションを選択して、**「完了」**をクリックします。"MapForceExtension" という名前の新規プロジェクトがワークプレイス内に作成されます。

サンプル拡張 プラグインの実行：

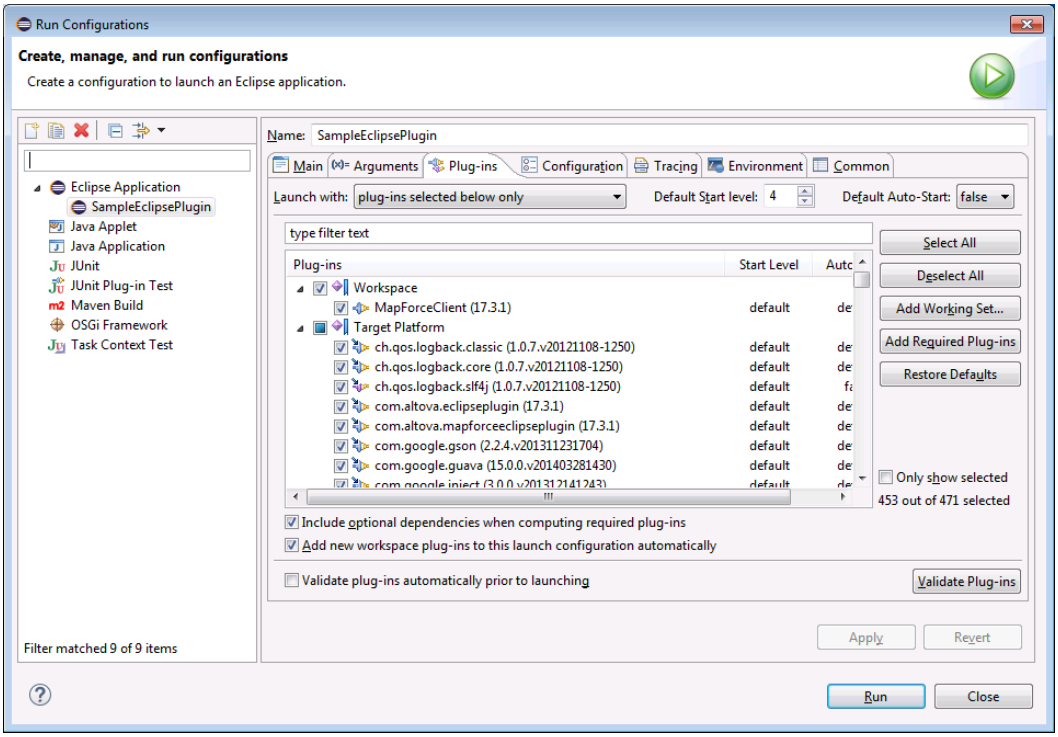
1. Java パースペクティブに切り替えます。
2. **「実行」**メニューから**「構成を実行」**をクリックします。
3. **「Eclipse アプリケーション」**を右クリックし、**「新規作成」**を選択します。(リスト内に "Eclipse アプリケーション"が存在しない場合、Eclipse プラグイン開発ツールがEclipse 環境にインストールされていないことを意味します。Eclipse プラグインプラグイン開発ツールをインストールするには、**「ヘルプ」**メニュー内の**「新しいソフトウェアをインストールする」**をクリックして、"Eclipse プラグイン開発ツール"を "Eclipse プロジェクト更新 "ダウンロードサイトからインストールします。



4. 名前と新しい構成を入力し、(このサンプルでは、*SampleMapForcePlugin*) **「適用」**をクリックします。



5. **MapForceClient** ワークプレイス プラグインが 'プラグイン' タブ内で選択されていることを確認してください。



6. **実行**をクリックして、新しいEclipse ワークベンチが開かれます。
7. MapForce マッピングを新しいワークベンチで開きます。マッピングは、70%のズームレベルで開かれます。

Chapter 15

メニューレファレンス

15 メニューレファレンス

以下のセクションでは、MapForce に用意されているメニューやメニューオプションのリストと、それらの簡単な説明を記します。

15.1 ファイル

新規作成
新規マッピングドキュメントまたはマッピングプロジェクト (*.mfp) を開きます。

開く
既に保存されたマッピング (*.mfd) またはマッピングプロジェクト (*.mfp) ファイルを開きます。使用中のMapForce エディションで使用できない機能が含まれているファイルは、開くことができません・

上書き保存
現在アクティブなファイル名で、現在アクティブになっているマッピングを保存します。

名前をつけて保存
現在アクティブになっているマッピングを、別の名前で保存します。また保存されていないファイルの場合、ファイルに新たな名前をつけることができます。

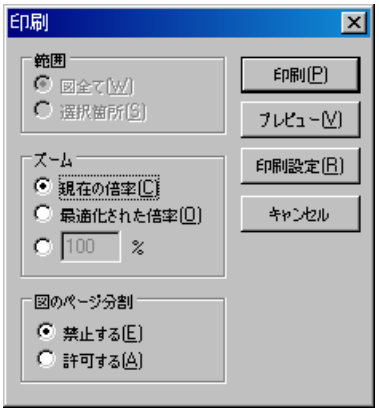
全て保存
現在開かれている全てのマッピングファイルを保存します。

再ロード
現在アクティブになっているマッピングを再ロードします。それまでに作成された変更点を破棄してもよいか尋ねられます。

閉じる
現在アクティブになっているマッピングを閉じます。ファイルを閉じる前に保存するか尋ねられます。

全て閉じる
現在開かれている全てのマッピングファイルを閉じます。保存されていないファイルを保存するか尋ねられます。

印刷
印刷ダイアログボックスが開かれ、マッピングを印刷することができます。



印刷ダイアログボックス

「現在の倍率」を選択すると、現在マッピングに対して指定されているズーム倍率が使用されます。**「最適化された倍率」**を選択すると、マッピングがページの大きさに収められるよう、印刷が最適化されます。コンポーネントのスクロールバーは印刷されません。1つのコンポーネントが複数のページに切り分けて印刷してもよいか指定することもできます。

印刷プレビュー
上に記述された機能を持った印刷ダイアログボックスが表示されます。

印刷設定
印刷設定ダイアログボックスが表示され、使用するプリンターや用紙の設定などを指定することができます。

マッピングの検証
マッピングを検証により、全てのマッピング（接続）が妥当であるかと、エラーや警告が表示されます。
詳細についてはマッピングの検証を参照ください。

マッピングの設定
マッピングの設定ダイアログボックスではドキュメントごとの設定を定義することができます（マッピングの設定の変更 を参照してください）。

選択された言語でコードを生成する
現在選択されているマッピングの言語によりコードが生成されます。現在選択されている言語は、ツールバーでハイライトされているプログラミング言語のアイコン（XSLT、XSLT2、XQuery、Java、C#、またはC++）で確認することができます。

コード生成 | XSLT (XSLT2)
このコマンドにより、ソースファイルからの変換に必要なXSLT ファイルが生成されます。このオプションを選択すると、フォルダーの参照ダイアログボックスが表示され、生成されたXSLT ファイルを配置する場所を指定することができます（マッピング設定の変更 を参照してください）。

コード生成 | XQuery
このコマンドにより、ソースファイルからの変換に必要なXQuery ファイルが生成されます。このオプションを選択すると、フォルダーの参照ダイアログ ボックスが表示され、生成されたXQuery ファイルを配置する場所を指定することができます（マッピング設定の変更 を参照してください）。

コード生成 | Java | C# | C++
これらのコマンドにより、ソースファイルから変換を行うのに必要なアプリケーションプログラムのコードが生成されます。このコマンドを選択すると、フォルダーの参照ダイアログボックスが表示され、生成されたファイルが配置される場所を指定することができます。生成されるアプリケーションファイル（ならびにプロジェクトファイル：*csproj C# プロジェクトファイル、*sln ソリューションファイル、*vcproj Visual C++ プロジェクトファイル）の名前は、マッピング設定ダイアログボックスのアプリケーション名フィールドにより定義されます。メニューオプションの「ファイル| マッピングの設定」を選択することで、このダイアログへアクセスすることができます。（マッピング設定の変更 を参照してください）。

ターゲットがXML/スキーマドキュメントの場合、コードの実行により作成されるファイル名は、コンポーネント設定ダイアログボックスの**出力 XML ファイル**フィールドにて与えられている名前となります。

MapForce Server 実行ファイルのコンパイル
マッピングの変換を実行するためにMapForce Server で実行されるファイルを生成します（MapForce マッピングのコンパイル を参照してください）。

FlowForce Server へデプロイ
現在アクティブなマッピングをFlowForce Server へデプロイします（MapForce マッピングのデプロイ を参照してください）。

ドキュメントの生成
様々な出力フォーマットにより、マッピングプロジェクトの詳細なドキュメンテーションを生成します（マッピングドキュメントの生成とカスタム化 を参照してください）。

最近のプロジェクト
最近開かれたプロジェクトのリストを表示します。

終了
アプリケーションを終了します。保存されていないドキュメントがある場合、それらを保存するか尋ねられます。

15.2 編集

このメニュー以下にあるコマンドの殆どは、**出力** タブにてマッピングの結果を参照している時、またはXSLT タブにてXSLT コードのプレビューを行なっている際に利用することができます。

元に戻す
MapForce では使用回数に制限の無い「元に戻す」操作を行うことができ、マッピングのデザインステップを順を追って確認することができます。

やり直し
やり直しコマンドにより、元に戻すコマンドにより戻された動作を進めることができます。これらのコマンドを使用することで、操作の履歴を確認することができます。

検索
XSLT、XSLT2、、XQuery または出力タブにて、指定されたテキストを検索することができます。

次を検索 F3
同じ検索文字列が次に出現する箇所を検索することができます。

前を検索 Shift F3
同じ検索文字列が前に出現する箇所を検索することができます。

切り取り/コピー/貼り付け/削除
標準的なWindows の編集コマンドです。マッピングウィンドウ内に表示されているコンポーネントや関数を切り取ったりコピーすることができます。

すべて選択
マッピングタブにある全てのコンポーネントを選択する、またはXSLT、XSLT2、XQuery XQuery、出力タブに表示されているテキスト/コード全体を選択することができます。

15.3 挿入

XML スキーマ / ファイル 

XML スキーマまたはインスタンスをマッピングに追加します。スキーマを参照するXML ファイルを選択すると、追加情報をマッピングに追加する必要はありません。スキーマ参照のないXML ファイルを選択すると、自動的に一致するXML スキーマを生成するかを問われます。 (XML スキーマの生成 を参照してください)。XML スキーマファイルを選択すると、プレビューのためのデータを提供するXML インスタンスを任意で追加するかを問われます。

データベース 

データソースまたはターゲットコンポーネントとしてデータベースコンポーネントを挿入することができます。データベースによりスキーマ構造が与えられ、ツリー形式により表示されます。

EDI 

データソースまたはターゲットコンポーネントとして使用することができるEDI ドキュメントを挿入することができます。詳細についてはEDI のセクションを参照ください。

テキストファイル 

フラットなテキストファイルドキュメント (例：CSV)、または固定長のテキストファイルを挿入することができます。両方の種類とも、ソースならびにターゲットコンポーネントとして使用することができます。更に、CSV または固定長とは異なる構造でテキストファイルを処理する場合、を使用することができます (MapForce FlexText を参照してください)。

Web サービス関数 

WSDL ファイルにより定義されたWeb サービス関数を挿入します。入力ならびに出力コンポーネントへ接続することで、出力ウィンドウにてWeb サービス関数の結果を素早く確認することができます。(詳細については" Web サービスの呼び出し" を参照ください。)

Excel 2007+ ファイル 

Microsoft Excel 2007+ (.xlsx) ファイルをマッピングに追加します (Microsoft OOXML Excel 2007+ を参照)。If you don't have Excel 2007 を使用していない場合、Excel 2007+ からマップ または Excel 2007+ へマップすることができます。この場合、出力タブで結果をプレビューすることはできませんが、**出力**メニューの**出力ファイルの保存**をクリックしてファイルを保存することができます。

XBRL ドキュメント 

XBRL インスタンスまたはタクソノミドキュメントを挿入します。タクソノミならびにインスタンスファイルのマッピングに関する詳細については、XBRL マッピング を参照ください。

JSON スキーマ/ファイル 

JSON スキーマまたはファイルを挿入します。詳細に関しては、JSON ファイルをマッピングコンポーネントとして追加するを参照してください。

入力の挿入 

マッピングウィンドウがマッピングを表示すると、このコマンドにより入力コンポーネントをマッピングに追加することができます。簡単な入力を参照。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドはユーザー定義関数の入力コンポーネントを追加します。複雑な入力コンポーネントの定義を参照。

出力の挿入 

マッピングウィンドウがマッピングを表示すると、このコマンドにより出力コンポーネントをマッピングに追加することができます簡単な出力を参照。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドはユーザー定義関数の出力コンポーネントを追加します。複雑な出力コンポーネントの定義を参照。

定数 

常に同じ値を入力アイコンへ与える関数コンポーネントの定数を挿入します。定数コンポーネントを作成する際に表示されるダイアログボックスにてデータを入力します。定数関数には出力アイコンが 1 つだけ含まれています。データの型は、文字列、数値、その他から選択することができます。

変数 

標準的な (非インライン型の) ユーザー定義関数と等価な中間変数を挿入します。変数はインスタンスファイルを持たない構造コンポーネントで、マッピング処理を単純にするために使用されます。詳細については 中間変数 を参照ください。

並べ替え:node/row

ノードの並べ替えを許可するコンポーネントを挿入します。詳細に関しては、 ノード行の並べ替えを参照してください。

フィルター:node/row 

node/ row と**bool**、そして**on- true** と**on- false** という 2 つの入力と出力パラメーターを備えたコンポーネントを挿入します。Boolean が true となっている場合、node/row パラメーターの値が on-true パラメーターから渡され、Boolean が false となっている場合、node/row パラメーターの値が on-false パラメーターへ渡されます。フィルターの使用方法については フィルターと条件を参照してください。

SQL-WHERE/ORDER 

データベースのデータに対して使用することができるフィルターコンポーネントを挿入することができます。MapForce により生成されたクエリに対して、任意の SQL WHERE 句を追加することができます。詳細については SQL WHERE / ORDER コンポーネント を参照ください。

Value-Map 

ルックアップテーブルを使って入力された値を特定の値へ変換するコンポーネントを挿入します。コンポーネントには 1 つの入力ならびに出力アイテムが与えられています。詳細については Value-Map - 入力データの変換 を参照ください。

IF-Else 条件 

"If-Else 条件 "のかたのコンポーネントを挿入します。詳細に関しては、 フィルターと条件を参照してください。

例外 

例外コンポーネントによりマッピングの処理を、特定の条件が満たされた場合に中断することができます。WSDL マッピングプロジェクトを使用してフォールトメッセージを定義することができます。 例外の追加 とWeb サービスのフォールトを参照してください。

15.4 プロジェクト

MapForce では、マルチドキュメントインターフェースがサポートされており、複数のマッピングを 1 つのマッピングプロジェクトへまとめることができます。詳細に関しては、マッピングプロジェクトとの作業 を参照してください。

プロジェクトの再ロード
現在アクティブなプロジェクトを再ロードして、プロジェクトタブを表示します。

プロジェクトを閉じる
現在アクティブになっているプロジェクトを閉じます。

プロジェクトの保存
現在アクティブになっているプロジェクトを保存します。

プロジェクトへファイルを追加する
ファイルを開くダイアログボックスにより、現在開かれているプロジェクトにマッピングを追加することができます。

アクティブなファイルをプロジェクトへ追加する
現在アクティブなファイルを、現在開かれているプロジェクトに追加することができます。

フォルダーの作成
このオプションにより、現在開かれているプロジェクト構造へ、新たなフォルダーが追加されます。このメニューオプションは、使用できるときだけ選択可能になります。詳細に関しては、プロジェクトフォルダーの管理を参照してください。

マッピングを開く
プロジェクトタブにて現在ハイライト選択されているマッピングを開きます。

オペレーションのためにマッピングを作成する
現在選択されているWSDL プロジェクトのオペレーションに対してマッピングを作成します。名前を付けて保存ダイアログボックスが自動的に表示され、WSDL ファイル内で定義されているオペレーション名が与えられます

オペレーションにマッピングファイルを追加する
既に保存されているマッピングファイルを現在アクティブなWSDL オペレーションへ追加することができます。ファイルを開くダイアログボックスからマッピングファイルを選択して下さい。

Web サービスの挿入 ...
既存のWSDL ファイルをベースにWeb サービスを挿入することができます。

XMLSpy のファイルを開く
XMLSpy のプロジェクトウィンドウで選択されているWSDL ファイルをXMLSpy で開きます。

プロジェクト全体のコードの生成
プロジェクトウィンドウに現在表示されているプロジェクト全体に対してプロジェクトコードを生成します。現在選択されているデフォルトの言語で、各フォルダーにある *mfd マッピングファイルのコードが生成されます。

コードの生成 ...
サブメニューから選択した言語にてプロジェクトコードを生成します。

プロパティ
プロジェクト全体の設定を定義することのできるダイアログボックスを開きます。詳細に関しては、コード生成設定の設定を参照してください。

最近のプロジェクト - 1 . 2 .etc.
最も最近に開かれたプロジェクトをリストします。

15.5 コンポーネント

ルート要素の変更
XML インスタンスドキュメントのルート要素を変更することができます。

XMLSpy でスキーマ定義を編集
XML スキーマコンポーネントが選択された状態でこのオプションを選択すると、XMLSpy のスキーマビューで XML スキーマファイルが開かれ、編集を行うことができます

FlexText 構成の編集
FlexText を開き、選択されている FlexText ファイルを編集することができます。

データベースオブジェクトの追加 /削除 /編集
データベースコンポーネント内のデータベースオブジェクトを追加、削除または変更することができます。

EDI X12 997 へのマッピングを作成
X12 997 Functional Acknowledgement により、EDI インターチェンジのステータスが報告されます。ドキュメントを処理する際に発生したエラーは全て、この中で報告されます。X12 997 受信確認 を参照ください。

EDI X12 999 へのマッピングを作成
X12 999 Implementation Acknowledgment Transaction Set により、HIPAA 実装ガイド不順守、またはアプリケーションエラーが報告されます。企業などの組織に送信された各 EDI トランザクションは、999 トランザクションを送信することで応答する必要があります。詳細については X12 999 - Implementation Guide non-compliance を参照ください。

更新
現在アクティブになっているデータベースコンポーネントの構造を、データベースから再ロードします。

前に入力を複製して追加
選択されたアイテムのコピー /クローンを、現在選択されているアイテムの前に挿入します。コピーされたアイテムには出力アイコンが含まれておらず、データ構造として使用することはできません。この機能を使用した例については 複数のソースから1つのターゲットにマップ セクションを参照ください。複製されたアイテムを右クリックすると、コンテキストメニューから複製アイテムを移動することができます。必要に応じて複製されたアイテムを上 /下へ移動させてください。

後に入力を複製して追加
選択されたアイテムのコピー /クローンを、現在選択されているアイテムの後に挿入します。コピーされたアイテムには出力アイテムが含まれておらず、データ構造として使用することはできません。この機能を使用した例については 複数のソースから1つのターゲットにマップ セクションを参照ください。複製されたアイテムを右クリックすると、コンテキストメニューから複製アイテムを移動することができます。必要に応じて複製されたアイテムを上 /下へ移動させてください。

複製を削除
既に作成された複製アイテムを削除します。この機能を使用した例については 複数のソースから1つのターゲットにマップ セクションを参照ください。

データベーステーブルのアクション
選択されているターゲットデータベーステーブルへ、データがマッピングされた際に取られるアクションを定義することができます。詳細については データベーステーブルアクションの設定 のセクションを参照ください。

データベースのクエリ
データベースコンポーネント内で選択されているテーブル /フィールドをベースに、SELECT ステートメントを作成します。テーブル /フィールドが選択されている状態でこのコマンドが使用可能になり、

SELECT ステートメントが SELECT ウィンドウに自動的に配置されます。

ツリーを左揃えにする
全てのアイテムをコンポーネントの左端へ揃えます。

ツリーを右揃えにする
全てのアイテムをコンポーネントの右端へ揃えます。この表示方法はターゲットスキーマへのマッピングを行う際に利用することができます。

プロパティ
現在選択されているコンポーネントの設定を表示するダイアログボックスを表示します。コンポーネント設定を変更するを参照してください。

15.6 接続

子要素の自動接続
子要素の自動接続機能を有効化または無効化します。ツールバーにあるアイコンからも選択を行うことができます。

子要素の接続設定
マッチする子要素の接続設定ダイアログボックスが表示され、接続設定を行うことができます。 (マッチする子要素の接続を参照)。

マッチする子要素の接続
このコマンドにより、ソースとターゲットスキーマ間で**同じ名前**を持った複数のアイテムに対して接続を作成することができます。このダイアログボックスで定義した設定はそのまま保持され、ツールバーにある**子要素の自動接続アイコン**  が有効になっている状態で 2 つのアイテムを接続した際に適用されます。このアイコンをクリックすることで、アイコンの有効/無効状態を切り替えることができます。詳細については マッチする子要素の接続のセクションを参照ください。

標準マッピング (ターゲット優先マッピング)
接続の種類を標準的なマッピングへ変更します。詳細については 標準マッピング (ターゲット優先マッピング) を参照してください。

全てコピー (子要素)
親コネクタのサブツリーとして表示されている子アイテムのコネクタが存在する状況で、マッチする全ての子要素に対してコネクタを作成します。詳細については すべてコピー接続 を参照ください。

ソース優先マッピング (混合コンテンツ)
接続をソース優先 (/混合コンテンツ)型のコネクタへ変更して、新たに要素を選択してマッピングすることができるようになります。詳細に関しては、 ソース優先マッピング (混合コンテンツ) を参照してください。

プロパティ
現在のコネクタに対して固有 (混合コンテンツ) の設定を定義するためのダイアログボックスが表示されます。利用することができないオプションはグレーアウトされます。これらの設定はテキストノードを持たない**複合型**のアイテムに対しても適用されます。詳細は接続設定を参照してください。

15.7 関数

ユーザー定義関数の作成：
新たなユーザー定義関数を作成します。 (詳細については ユーザー定義関数 を参照ください)。

選択からユーザー定義関数作成：
マッピングウィンドウにて現在選択されている要素をベースに、新たなユーザー定義関数を作成します。

関数設定：
現在アクティブなユーザー定義関数の設定ダイアログボックスを開き、現在の設定を変更することができます。

関数削除：
既存のユーザー定義関数を開き、その関数の名前がタブに表示されている状態で、現在アクティブなユーザー定義関数を削除します。

入力の挿入：
マッピングウィンドウがマッピングを表示すると、このコマンドは、マッピングに入力コンポーネントを追加します (詳細については 単純型入力 を参照ください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドは、マッピングに入力コンポーネントをユーザー定義関数に追加します (詳細については 複合型の入力コンポーネントの定義 を参照ください)。

出力の挿入：
マッピングウィンドウがマッピングを表示すると、このコマンドは、マッピングに出力コンポーネントを追加します (詳細については 単純型出力 を参照ください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドは、マッピングに出力コンポーネントをユーザー定義関数に追加します (詳細については 複合型の出力コンポーネントの定義 を参照ください)。

15.8 出力

(**XSLT 1.0、XSLT 2.0 などの**)最初のオプショングループにより、コードを生成する際の[ターゲット言語を定義](#) することができます。

出力ファイルの検証
参照されているスキーマに対して出力 XML ファイルを検証します。

生成された出力の保存
出力タブに現在表示されているデータを保存します。

生成されたすべての出力を保存
動的マッピングにより生成された出力をファイルに保存します。詳細については[複数の入力または出力ファイルを動的に処理](#) を参照ください。

出力を再生成
現在のマッピングを出力ウィンドウから再生成します。

SQL スクリプトを実行
SQL スクリプトが現在出力ウィンドウで表示されている場合、テーブルアクションの定義に従ったかたちで、ターゲットデータベースに対してマッピングを実行します。

ブックマークの挿入 /削除
出力ウィンドウのカーソル位置にブックマークを挿入します

次のブックマーク
出力ウィンドウ内にある次のブックマークへ移動します。

前のブックマーク
出力ウィンドウ内にある前のブックマークへ移動します。

全てのブックマークを削除
出力ウィンドウにて現在定義されている全てのブックマークを削除します。

XML テキストの整形
出力ペインに表示されている XML ドキュメントの表示を、ドキュメントの構造に従ったかたちに再構成します。各子ノードが、親ノードよりインデントされたかたちで表示されます。この機能では、タググループにて指定されたタブサイズの設定（タブまたはスペースを挿入）が使用されます。

テキストビューの設定
テキストビュー設定ダイアログボックスを表示します。このダイアログボックスにより**出力** ペイン、**XSLT** ペイン、と**XQuery** ペインでテキストビュー設定をカスタマイズすることができます。またこのダイアログボックスは、現在ウィンドウ内で適用されている定義済みのホットキーを表示しています。 詳細に関しては、次を参照してください[テキストビュー機能](#)。

15.9 デバッグ

デバッグの開始 (F11)

デバッグを開始または、ブレークポイントに到達またはマッピングが完了するまで継続します。

デバッグの停止 (Shift + F5)

デバッグを停止します。このコマンドはデバッグモードを終了して、MapForce を標準モードに切り替えます。

ステップイン (F11)

マッピング内で単一のステップを終了するまでマッピングを実行します。マッピングでデバッガーは、シーケンスの単一アイテムを生成する独立した計算の理論的なグループです。

マッピングコンテキストにより、このコマンドは "左へ移動 /ターゲットに移動 子 /ソースの親へ移動 " と解釈することができます。

ステップオーバー (F10)

現在のステップが終了するまで (または他のシーケンスのアイテムを終了するまで) 実行を継続します。このコマンドは、現在のステップの入力である計算をステップオーバーします。

ステップアウト (Shift + F11)

現在のステップの結果が使用されるまで、または入力または使用の子ではないステップが実行されるまで実行を継続します。このコマンドは現在の計算をステップアウトします。

マッピングコンテキストにより、このコマンドは "右へ移動 /ターゲットへ移動 親 /ソースの子へ移動 "と解釈することができます。

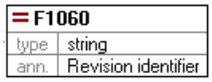
最小ステップ (Ctrl + F11)

値が生成または使用されるまで、実行を継続します。このコマンドは、ステップを細分化し、接続ごとに通常 2回停止します : ソースが値を生成した際、ターゲットが値を使用した際。MapForce は、マッピングの提示に従い値を計算する必要はなく、生成と使用イベントは順序に従わない場合があります。

15.10 表示

注釈の表示 

XML スキーマの注釈（ならびにEDI 情報）をコンポーネントウィンドウにて表示します。
型の表示アイコンも有効になっている場合、両方の情報がグリッド形式で表示されます。



型の表示 

各要素または属性に対してスキーマのデータ型を表示します。
注釈の表示アイコンも有効になっている場合、両方の情報がグリッド形式で表示されます。

関数ヘッダーにライブラリ名を表示 

関数タイトルは括弧付きでライブラリ名を表示します。

ヒントの表示 

マウスポインターが関数の上に配置された時に、ツールチップ内に関数の説明が表示されます。

XBRL 表示オプション

詳細に関しては、XBRL ラベル表示オプションの構成を参照してください。

選択されたコンポーネントの接続線の表示 

すべてのマッピングの接続線または、現在選択されているコンポーネントに関連した接続線を切り替えることができます。

ソースからターゲットへの接続線表示 

以下のように接続線の表示を切り替えることができます：

- 現在選択されているコンポーネントに**直接**接続されているコネクタ
- 現在選択されているコネクタで、ソースからターゲットコンポーネントまで

ズーム

ズームダイアログボックスが表示されます。ズームの倍率を手動で入力することができるほか、スライダーをドラッグして、ズームの倍率を変更することができます。

戻る

マッピングタブにて現在開かれているマッピングで、前に表示されていたマッピングを表示します

進む

マッピングタブにて現在開かれているマッピングで、戻るコマンドにより表示される前のマッピングを表示します。

ステータスバー

メッセージウィンドウの下に表示されているステータスバーを表示／非表示にします。

ライブラリウィンドウ

ライブラリ関数が含まれるライブラリウィンドウを表示／非表示にします。

メッセージウィンドウ

検証出力ウィンドウを表示／隠します。コードの生成を行う場合、メッセージ出力ウィンドウが自動的に有効になり、検証結果が表示されます。

概要ウィンドウ
概要ウィンドウを表示/非表示にします。長方形をドラッグすることで、マッピングの表示範囲を移動することができます

プロジェクトウィンドウ
プロジェクトウィンドウを表示/非表示にします。

15.11 ツール

グローバルリソース
グローバルリソースの管理ダイアログボックスを表示して、グローバルリソースXML ファイル内にあるグローバルリソース情報を追加、編集、または削除することができます。詳細については グローバルリソース - プロパティを参照ください。

アクティブな構成
現在アクティブになっているグローバルリソースを、グローバルリソースの構成リストから選択/変更することができます。サブメニューから目的の構成を選択してください。

反転マッピングの作成
MapForce で現在アクティブになっているマッピングをベースに、「反転した」マッピングを作成します。反転した結果生成されるマッピングは完成したものではなく、コンポーネント間における直接接続だけが保持されるという点に注意してください。恐らく反転したマッピングは妥当なものではなく、更なる編集を行うこと無く出力タブをクリックしても、実行されません。

マッピングを反転させると、ソースコンポーネントがターゲットコンポーネントになり、ターゲットコンポーネントがソースコンポーネントになります。入力または出力 XML インスタンスファイルがコンポーネントに割り当てられている場合、これらも交換されます。

以下のデータが保持されます：

- コンポーネント間の直接接続
- チェーンマッピング内のコンポーネント間の直接接続
- 接続の種類：標準、混合コンテンツ、全てコピー
- パススルーコンポーネント設定
- データベースコンポーネント

以下のデータが保持されません：

- 関数やフィルターなどを経由した接続は、関数などとともに削除されます
- ユーザー定義関数
- Web サービスコンポーネント

ツールバーとウィンドウの復元
ツールバーやドックウィンドウなどをデフォルトの状態にリセットします。変更を反映するには、MapForce を再起動する必要があります。

カスタム化 ...
MapForce グラフィカルユーザーインターフェイスをカスタム化することのできるダイアログボックスを開きます。これはツールバーの表示と非表示、またコンテキストメニューの編集とキーボードのショートカットを含みます（キーボードショートカットのカスタム化 を参照してください）。

オプション
デフォルトのMapForce 設定を変更することのできるダイアログボックスを開きます（MapForce オプションの変更 を参照してください）。

15.12 ウィンドウ

重ねて表示
開かれている全てのウィンドウを重ねて表示されるように再配置します。

上下に並べて表示
開かれている全てのウィンドウを**上下に並べて表示**することで、同時に表示されるように再配置します。

左右に並べて表示
開かれている全てのウィンドウを**左右に並べて表示**することで、同時に表示されるように再配置します。

1
2
現在開かれているウィンドウが表示され、これらウィンドウ間で素早い切り替えを行うことができます。
Ctrl + tab または Ctrl + F6 キーボードショートカットを使用することでも、開かれているウィンドウを切り替えることができます。

15.13 ヘルプメニュー

目次

説明

ヘルプウィンドウの左側のペインに目次を表示した、MapForce の画面上のヘルプマニュアルを開きます。目次はヘルプドキュメント全体の概要を表示しています。目次のエントリをクリックしてトピックに移動することができます。

インデックス

説明

ヘルプウィンドウの左側のペインにキーワード インデックスを表示したMapForce の画面上のヘルプマニュアルを開きます。目次はヘルプドキュメント全体の概要を表示しています。インデックスはキーワードをリストし、キーワードをダブルクリックすることでトピックへ移動することができます。キーワードが1つ以上のトピックにリンクされている場合は、トピックのリストが表示されます。

検索

説明

ヘルプウィンドウの左側のペインに検索ダイアログを表示したMapForce の画面上のヘルプマニュアルを開きます。単語を検索するには、入力フィールドに検索対象を入力して、**[Return]** を押します。ヘルプシステムは、ヘルプドキュメント全体で全文検索を行いヒットしたリストを返します。アイテムを表示するためにはアイテムをダブルクリックします。

ライセンス登録

説明

Altova 製品ソフトウェアをダウンロードすると、無料評価キーまたは購入されたライセンスキーを使用して、製品のアクティベーションを行うことができます。

- ***無料評価キー:*** 初めて製品のダウンロードとインストールを行うと、ソフトウェアアクティベーションダイアログが表示されます。ダイアログでは無料評価キーコードをリクエストすることができます。ユーザーの名前、所属会社名、そして電子メールアドレスを入力して、無料評価キーを取得するボタンをクリックしてください。入力された電子メールアドレスに評価キーが送信されます。ソフトウェアアクティベーションのダイアログボックスにで送信されたキーコードを入力し、**[OK]** をクリックすることで、Altova 製品を使用することができるようになります。ソフトウェアは30日の間アンロックされます。
- ***購入されたライセンスキー:*** ソフトウェアアクティベーションダイアログにはライセンスキーのキーコードを購入するボタンが含まれています。このボタンをクリックするとAltova のオンラインショップが表示され、製品に対応したライセンスキーを購入することができます。ライセンスにはシングルユーザーとマルチユーザーという二種類のライセンスがあります。両種類のキーコードが電子メールにより送信されます。シングルユーザーライセンスにはライセンスデータや組織名、そしてキーコードが含まれています。ライセンス契約では、ライセンスされている数を超えて組織内のコンピューターにAltova 製品をインストールできないということに注意してください。電子メール内に含まれているライセンス情報を、登録ダイアログに正確に入力するようご注意してくだ

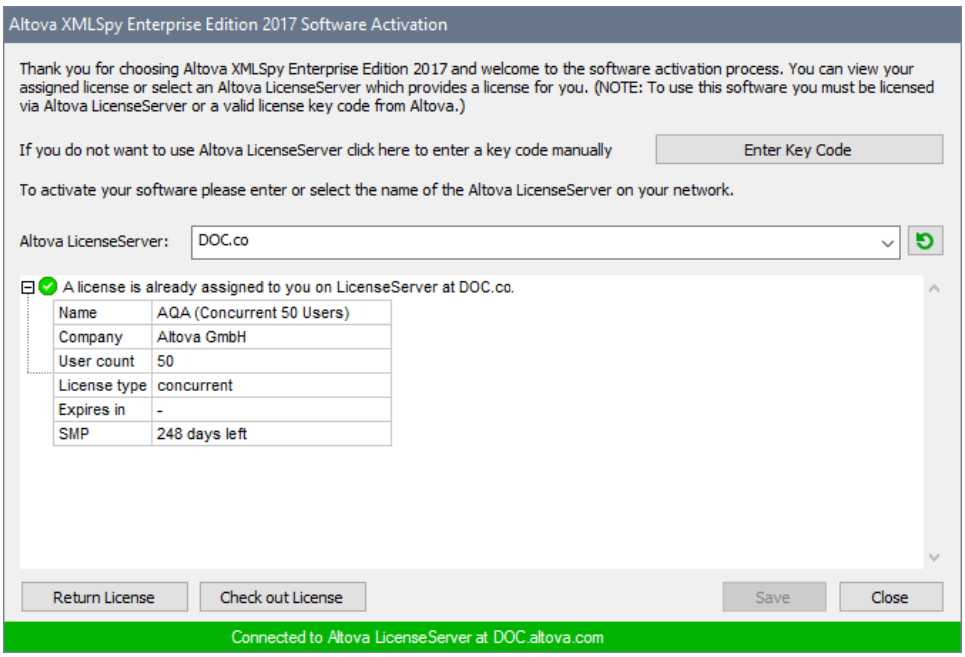
さい。

メモ: ソフトウェアアクティベーションダイアログにてライセンス情報を入力する際には、ライセンス電子メールに記された情報を正確に入力するようにしてください。マルチユーザーライセンスの場合、Name フィールドに、各ユーザーの名前を入力してください。

ソフトウェアアクティベーションダイアログ (*下のスクリーンショット*) は、**「ヘルプ | ソフトウェア アクティベーション」**をクリックすることにより常にアクセスすることができます。

以下の方法によりソフトウェアをアクティブ化することができます:

- (**「新しいキーコードを入力する」**をクリックして) ライセンスキーの情報を入力します。または、
- (**「Altova LicenseServer を使用」**をクリックして) ネットワーク上の Altova LicenseServer からライセンスを取得します。Altova LicenseServer は、使用中の Altova 製品のためのライセンスをライセンスプール上に有する必要があります。ライセンスが LicenseServer プール内に存在する場合、Software Activation ダイアログ内に表示されます (*下のスクリーンショット*)。**「保存」**をクリックしてライセンスを取得します。



ライセンスが1度取得されると、7日間は、LicenseServer に戻すことができません。7日過ぎると、(**「ライセンスを戻す」**をクリックして) ライセンスを戻すことができ、ライセンスは、他のクライアントから取得することができます。LicenseServer 管理者は、LicenseServer の Web UI を使用して、取得されたライセンスの割り当てを解除することができます。

ライセンスのチェックアウト

ライセンスが製品マシン上に保管されるように、ライセンスをライセンスプールから 30 日間チェックアウトすることができます。これにより、オフラインで作業することが可能になります。これはとても役に立ちます。Altova LicenseServer にアクセスできない環境 (例えば、旅行中に Altova 製品がインストールされたラップトップコンピューターで作業する) 場合などが挙げられます。ライセンスはチェックアウトされていますが、LicenseServer は、ライセンスが使用中と表示し、ライセンスは他のマシンで使用することができません。ライセンスはチェックアウトの期間が終わると自動的にチェックインされた状態を戻します。または、チェックアウトされたライセンスはソフトウェアのライセンスの認証ダイアログのボタンを使用して**チェックイン**することで

きます。

ライセンスをチェックアウトするには以下をおこないます：(i) ソフトウェアのライセンスの認証ダイアログで **ライセンスのチェックアウト** をクリックします（*上のスクリーンショット参照*）。 (ii) ライセンスのチェックアウトダイアログ内から、チェックアウトの期間を選択し、**チェックアウト**をクリックします。ライセンスがチェックアウトされます。ソフトウェアのライセンスの認証ダイアログは、チェックアウト期間の終わりを含むチェックアウトの情報を表示します。ダイアログ内の**ライセンスのチェックアウト** ボタンは、**チェックイン**ボタンに変わります。**チェックイン**ボタンをクリックして、ライセンスをチェックインすることができます。ライセンスは自動的にチェックイン状態に戻されるため、選択したチェックアウトの期間がオフラインで作業する期間をカバーするようご確認ください。

Altova LicenseServer を使用することにより、IT 管理者は、リアルタイムでネットワーク上の全てのライセンスの概要、および、クライアントの割り当てと、クライアントのライセンスの使用状況を確認することができます。LicenseServer を使用する利点は、ですから、多数の Altova ライセンスを管理することのできる管理機能です。 Altova LicenseServer は、Altova Web サイトで無料で提供されています。Altova LicenseServer および Altova LicenseServer を使用したライセンスの供与に関する詳細は、Altova LicenseServer ドキュメントを参照してください。

注文フォーム

説明

ソフトウェア製品のライセンスを購入する準備が整った場合、ソフトウェアアクティベーションダイアログにて **[キーコードを購入する]** ボタンを選択するか、**[ヘルプ | 注文フォーム]** コマンドを選択して、Altova オンラインショップへアクセスすることができます。

登録

説明

Altova 製品登録ページをブラウザーのタブに表示します。Altova ソフトウェアを登録することにより、最新の製品の情報が得られます。

最新情報のチェック

説明

Altova サーバーに接続して、より新しいバージョンの製品が利用可能かどうかチェックし、その結果を表示します。

サポートセンター

説明

インターネット上にある Altova サポートセンターへのリンクとなっています。サポートセンターには FAQ やディスカッ

ションフォーラムが含まれており、問題の解決方法を探ったり、Altova の技術サポートスタッフへアクセスすることができます（現在英語のみの提供となります）。

WEB 上の FAQ

説明

インターネット上にある Altova の FAQ へのリンクとなっています。FAQ データベースは Altova のサポートスタッフにより常時更新されています。

コンポーネントのダウンロード

説明

インターネット上にある Altova のコンポーネントダウンロードセンターへのリンクとなっています。このリンク先から様々なコンポーネントソフトウェアをダウンロードして、Altova 製品とともに使用することができます。ソフトウェアコンポーネントは XSLT や XSL-FO プロセッサーからアプリケーションサービスプラットフォームまで、幅広く提供されています。コンポーネントダウンロードセンターにてご利用になれるソフトウェアは、通常無料でご利用になれます。

インターネット上の MapForce

説明

インターネット上にある Altova ウェブサイトへのリンクとなっています。Altova ウェブサイトでは、MapForce や関連するテクノロジーについて確認することができます。

MapForce について

説明

スプラッシュ画面と製品のバージョン番号が表示されます。IMapForce の 64 ビットバージョンを使用している場合、アプリケーション名の後のサフィックス (x64) を示しています。32 ビットバージョンにはサフィックスはありません。

Chapter 16

Code Generator

16 Code Generator

Code Generator is a MapForce built-in feature which enables you to generate Java, C++ or C# code from mapping files designed with MapForce. You can generate code not only from simple mappings with a single data source and target, but also from mappings with multiple sources and multiple targets. The result is a fully-featured and complete application which performs the mapping operation for you. Once you generate the code, you can execute the mapping by running the application directly as generated. You can also import the generated code into your own application, or extend it with your own functionality.

If your mapping uses XML schemas or DTDs, you can also optionally generate schema wrapper libraries (see Generating Code from XML Schemas or DTDs). The schema wrapper libraries enable you to work with XML data in an abstract way, without too much concern for the underlying XML Application Program Interface (API), such as MSXML, Apache Xerces, Microsoft System.Xml, or Java Application for XML Processing (JAXP).

(C) 2017 Altova GmbH

16.1 Introduction to code generator

The primary goal of the generated code is to execute a MapForce mapping. In addition to this, you can optionally generate schema wrapper libraries for XML schemas used by the mapping, which enables you to read or write data to/ from XML instances.

The generated code is expressed in C++, Java or C# programming languages.

Target Language	C++	C#	Java
Development environments	Microsoft Visual Studio 2010 Microsoft Visual Studio 2008 Microsoft Visual Studio 2005	Microsoft Visual Studio 2010 Microsoft Visual Studio 2008 Microsoft Visual Studio 2005	Eclipse 3.4 or later Apache Ant (build.xml file)
XML DOM implementations	MSXML 6.0 or Apache Xerces 2.6 or later	System.Xml	JAXP
Database API	ADO	ADO.NET	JDBC

C++

You can configure whether the C++ generated output should use MSXML 6.0 or Apache Xerces 2.6 or later. MapForce generates complete project (.vcproj) and solution (.sln) files for Visual Studio 2005/ 2008/ 2010. The generated code optionally supports MFC.

Note: When building C++ code for Visual Studio 2005/2008/2010 and using a Xerces library precompiled for Visual C++, a compiler setting has to be changed in all projects of the solution:

1. Select all projects in the Solution Explorer.
2. On the **Project** menu, click **Properties**.
3. Click **Configuration Properties | C/C++ | Language**.
4. In the list of configurations, select *All Configurations*.
5. Change **Treat wchar_t as Built-in Type** to **No (/ Zc:wchar_t-)**

C#

The generated C# code uses the .NET XML classes (System.Xml) and can be used from any .NET capable programming language, such as VB.NET, Managed C++, or J#. Project files can be generated for Visual Studio 2005, 2008, and 2010.

Java

The generated Java output is written against the industry-standard Java API for XML Processing (JAXP) and includes an Ant build file and project files for Eclipse. Java 7 or higher is supported.

Generated output

The designated destination folder will include all the libraries and files required to execute the mapping, namely:

- A variable number of Altova libraries required by the mapping (for example, Altova function

libraries, database libraries, EDI libraries)
- A complete mapping application. When compiled and run, the application performs the mapping transformation.

Code generator templates

Output code is completely customizable via a simple yet powerful template language (SPL, from Spy Programming Language) which gives full control in mapping XML Schema built-in data-types to the primitive datatypes of a particular programming language. SPL allows you to easily replace the underlying parsing and validating engine, customize code according to your company's writing conventions, or use different base libraries such as the Microsoft Foundation Classes (MFC) and the Standard Template Library (STL).

(C) 2017 Altova GmbH

16.2 What's new ...

Version 2014

- Removal of compatibility mode option for code generation

Version 2011

- Contains bug fixes and enhancements

Version 2010 R3

- Support for Microsoft Visual Studio 2010
- Support for MSXML 6.0 in generated C++ code
- Support for 64-bit targets for C++ and C# projects

Version 2010

- Enumeration facets from XML schemas are now available as symbolic constants in the generated classes (using 2007r3 templates)

Version 2009 sp1

- Apache Xerces version 3.x support added (older versions starting from Xerces 2.6.x are still supported)

Version 2009

- The generated mapping implementation was redesigned to support sequences and grouping. The API has not changed

Version 2008 R2

- Support for generation of Visual Studio 2008 project files for C# and C++ has been added
- Generated MapForce mapping code in C# and Java can use readers/writers, streams, strings or DOM documents as sources and targets

Version 2008

The new 2007 R3-style SPL templates have been further enhanced:

- It is now possible to remove single elements
- Access to schema metadata (e.g. element names, facets, enumerations, occurrence, etc.) is provided
- Complex types derived by extension are now generated as derived classes

Version 2007 R3

Code Generator has been redesigned for version 2007 release 3 to simplify usage of the generated code, reduce code volume and increase performance.

- Handling of XML documents and nodes with explicit ownership, to avoid memory leaks and to enable multi-threading
- New syntax to avoid name collisions
- New data types for simpler usage and higher performance (native types where possible, new null handling, ...)
- Attributes are no longer generated as collections
- Simple element content is now also treated like a special attribute, for consistency
- New internal object model (important for customized SPL templates)
- Compatibility mode to generate code in the style of older releases
- Type wrapper classes are now only generated on demand for smaller code

(C) 2017 Altova GmbH

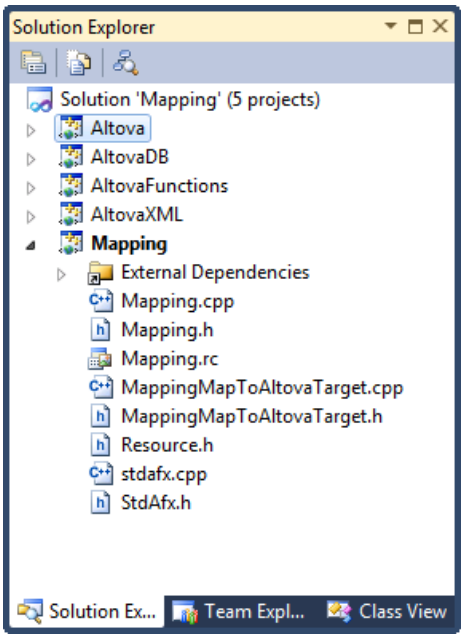
16.3 Generating C++ code

You can generate C++ code for Visual Studio 2005, 2008 and 2010. The generated code includes **.sln** and .**vcproj** files for Visual Studio. Note the following when generating code:

- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**.

A typical C++ solution generated by MapForce includes the following:

- Several Altova-signed libraries required by the mapping (all prefixed with **Altova**).
- The main mapping project (in this example, **Mapping**), which includes the mapping application and dependent files.



Sample C++ solution generated with MapForce

This section includes the following topics:

- Generating code from a mapping
- Generating code from a mapping project
- Building the project
- Running the application

16.3.1 Generating code from a mapping

To generate C++ code from a mapping design file (.mfd):

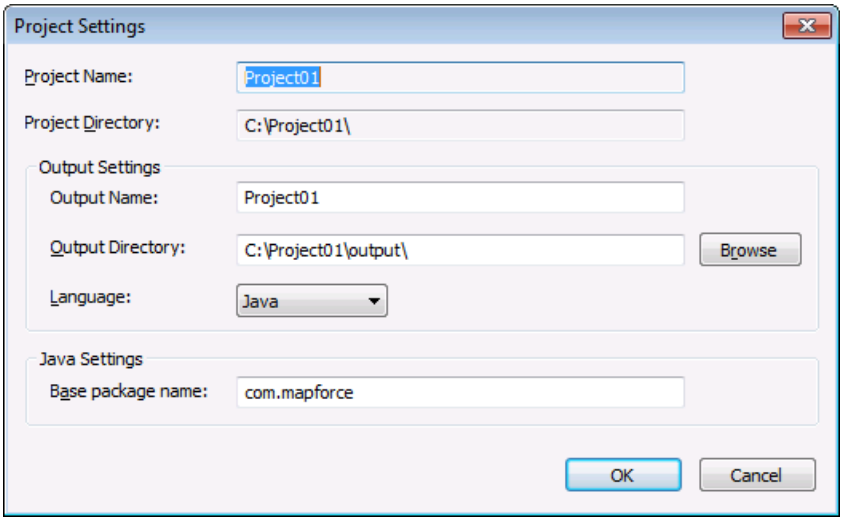
1. Review and select the desired code generation options (see Code generator options).
2. On the **File** menu, click **Generate code in | C++**.
3. Select a destination directory for the generated files, and then click OK to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**. If required, you can change this, and other settings, from the Mapping Settings dialog box (see Changing the mapping settings).

16.3.2 Generating code from a mapping project

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project** menu, click **Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

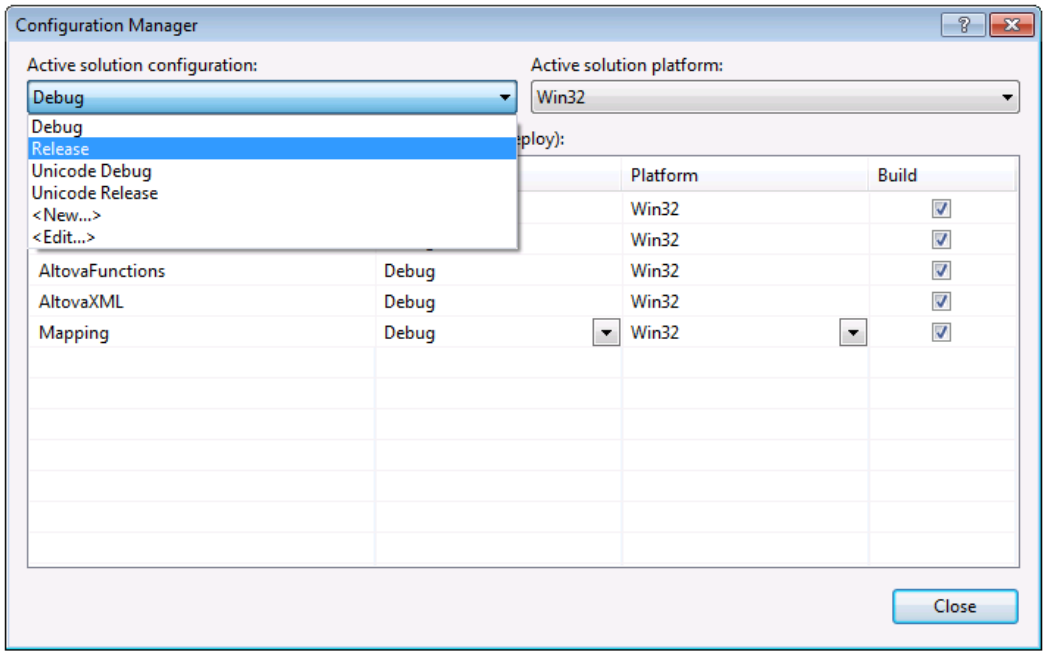
16.3.3 Building the project

Once you generated the C++ code, building it in Visual Studio is the next step. To build the generated code:

1. Open the generated solution (.sln) file in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.

2. On the **Build** menu, click **Configuration Manager**.



3. Select the required build configuration (Debug, Release, Unicode Debug, Unicode Release). Note that only Unicode builds support the full Unicode character set in XML and other files. The non-Unicode builds work with the local codepage of your Windows installation.
4. On the **Build** menu, click **Build Solution**.

16.3.4 Running the application

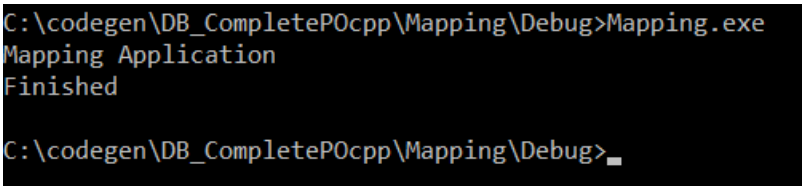
Once you compile the Visual Studio project, a command-line application is produced, called **Mapping.exe**. (Note that if you changed the application name from the mapping settings, then the executable name is changed accordingly.)

You can locate the mapping application in one of the following subdirectories relative to the .sln file, depending on the build option you chose:

- Debug

- Release
- Unicode Debug
- Unicode Release

To run the application, open a command prompt, change the current directory to the path of the executable, and run it, for example:



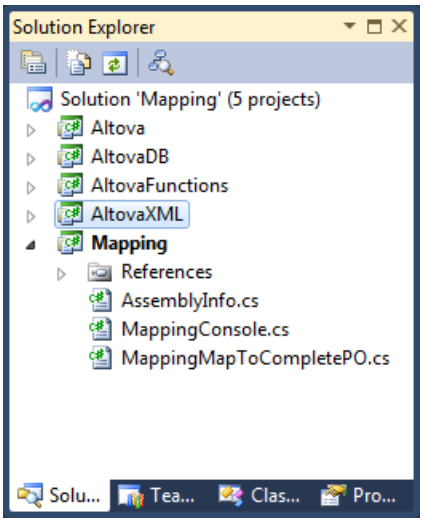
16.4 Generating C# code

You can generate C# code for Visual Studio 2005, 2008 and 2010. The generated code includes **.sln** and .**csproj** files for Visual Studio. Mono users can use Visual Studio solution files with Mono's xbuild. Note the following when generating code:

- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**.

A typical C# solution generated by MapForce includes the following:

- Several Altova-signed libraries required by the mapping (all prefixed with **Altova**).
- The main mapping project (in this example, **Mapping**), which includes the mapping application and dependent files.



Sample C# solution generated with MapForce

This section includes the following topics:

- Generating code from a mapping
- Generating code from a mapping project
- Building the project
- Running the application

16.4.1 Generating code from a mapping

To generate C# code from a mapping design file (.mfd):

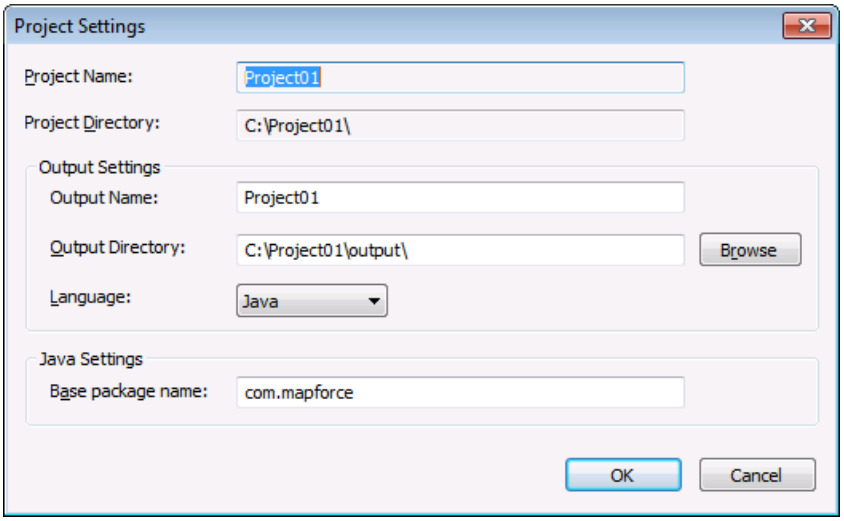
1. Review and select the desired code generation options (see Code generator options).
2. On the **File** menu, click **Generate code in | C#**.
3. Select a destination directory for the generated files, and then click OK to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**. If required, you can change this, and other settings, from the Mapping Settings dialog box (see Changing the mapping settings).

16.4.2 Generating code from a mapping project

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project** menu, click **Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

16.4.3 Building the project

Once you generated the C# code, building it in Visual Studio is the next step. To build the generated code:

1. Open the generated solution (.sln) file in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.

2. On the **Build** menu, click **Configuration Manager**.
3. Select the required build configuration (Debug, Release).
4. On the **Build** menu, click **Build Solution**.

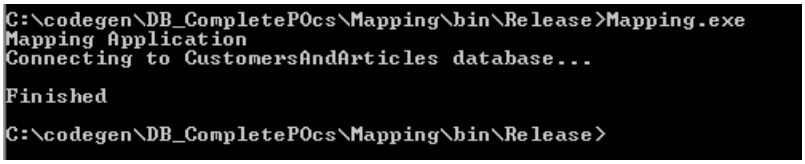
16.4.4 Running the application

Once you compile the Visual Studio project, a command-line application is produced, called **Mapping.exe**. (Note that if you changed the application name from the mapping settings, then the executable name is changed accordingly.)

You can locate the mapping application in one of the following subdirectories relative to the .sln file, depending on the build option you chose:

- bin\Debug
- bin\Release

To run the application, open a command prompt, change the current directory to the path of the executable, and run it, for example:



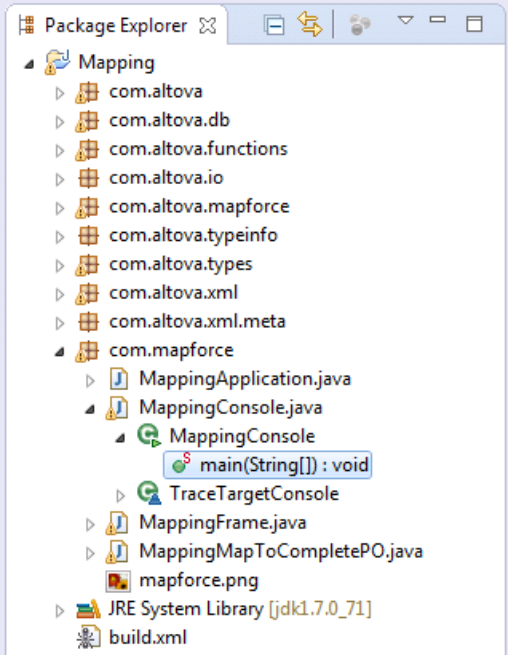
16.5 Generating Java code

You can generate program code for Java version 7 or later. Note the following when generating code:

- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**.

A typical Java project generated by MapForce includes the following:

- Several Altova-signed Java packages required by the mapping (all prefixed with **com.altova**).
- The **com.mapforce** package, which includes the mapping application and dependent files (as shown below, it is possible to change the name of this package). The two most important files in this package are as follows:
  - The Java mapping application as a dialog application (**MappingApplication.java**).
  - The Java mapping application as a console application (**MappingConsole.java**).
- A **build.xml** file which you can execute with Apache Ant to compile the project and generate JAR files.



Sample MapForce-generated Java application (Eclipse IDE)

This section includes the following topics:

- Generating code from a mapping
- Generating code from a mapping project
- Handling JDBC references
- Building the project with Ant
- Example: Run and compile Java code with Eclipse and Ant

16.5.1 Generating code from a mapping

To generate Java code from a mapping design file (.mfd):

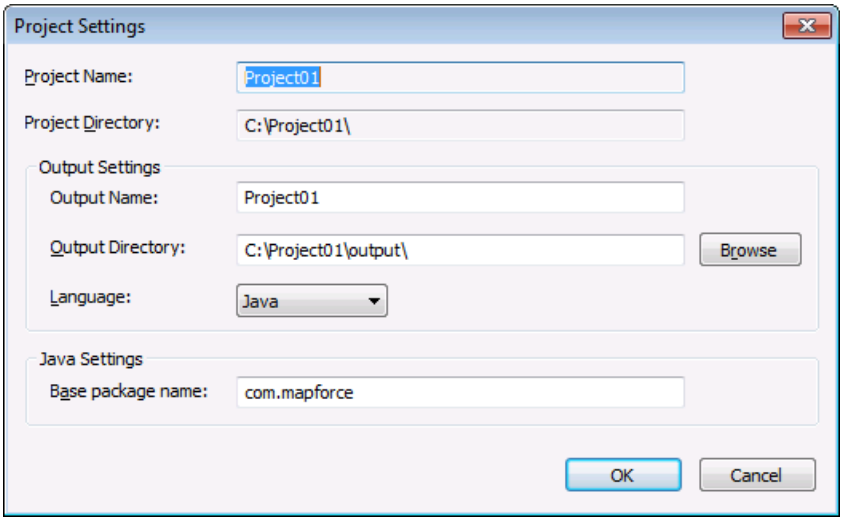
1. Review and select the desired code generation options (see Code generator options).
2. On the **File** menu, click **Generate code in | Java**.
3. Select a destination directory for the generated files, and then click OK to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**, and the default name of the base package is **com.mapforce**. If required, you can change these from the Mapping Settings dialog box (see Changing the mapping settings).

16.5.2 Generating code from a mapping project

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project** menu, click **Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

16.5.3 Handling JDBC references

If the mapping connects to a database through JDBC, ensure that the JDBC drivers used by the mapping are installed on your system (see Creating a JDBC connection). To view the current JDBC settings of any database component, click it, and then select **Component | Properties** from the menu.

Additionally, if you build JAR files from generated Java code, add the "Class-Path" attribute for your database driver to the **build.xml** file. This ensures that the reference to the database driver is available in the manifest (**MANIFEST.MF**) file after you build the project.

To add the "Class-Path" attribute:

1. Add to the **build.xml** file a reference to the JAR file of the database driver, as a new "Class-Path" attribute. For example, for MySQL 5.1.16, the value of the "Class-Path" attribute looks as follows:

```
<attribute name="Class-Path" value="mysql-connector-java-5.1.16-
bin.jar"/>
```

The **manifest** element of the **build.xml** file now looks similar to the screen shot below.

```
<manifest file="U:\_TTP_Verification__related\42730__.__SAP_iDoc\CODE_Jav
  <attribute name="Created-By" value="MapForce 2014"/>
  <attribute name="Main-Class" value="com.mapforce.Paccar866DBDELFOR_
  <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
</manifest>
```

2. Copy the JAR file of the JDBC driver to the folder that contains the JAR file of the generated application.

16.5.4 Building the project with Ant

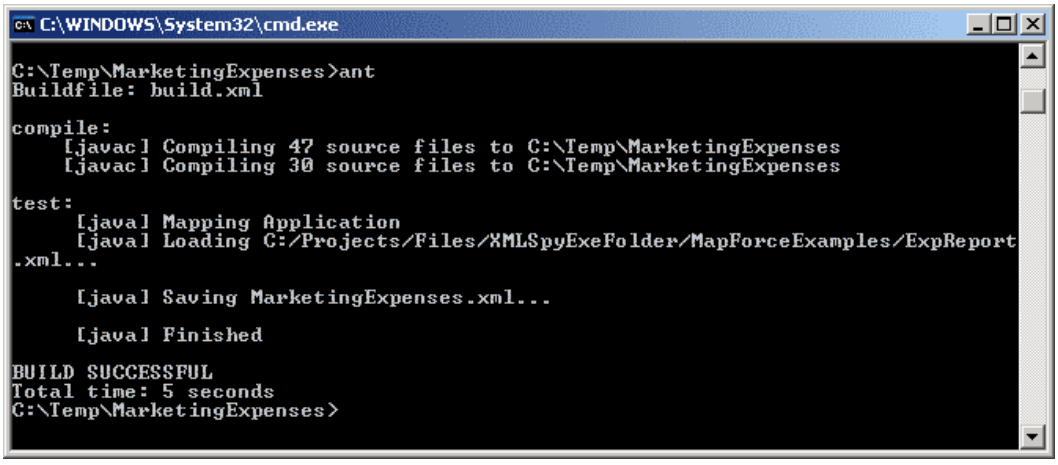
Apache Ant is a widely used Java library (and command-line tool) which automates building and compilation of Java projects (see http://ant.apache.org/manual/). Ant works with build files (such files define the sources and targets from which code must be compiled, as well as any specific build options). Since any MapForce-generated project includes a **build.xml** file recognized by Ant, you can easily build MapForce-generated projects with Ant.

Ant may be available on your system either as a standalone installation, or bundled within Eclipse (or other Java IDEs). For instructions on how to install Ant on your system, see http://ant.apache.org/manual/. For instructions on how to use Ant in Eclipse, refer to the Eclipse tutorial, as well as the Eclipse documentation.

You can quickly check whether the standalone version of Ant (not the one bundled with Eclipse) is available on your system by opening a command prompt and typing `ant` at the command line. When Ant is already available, the resulting message will be similar to: `Buildfile: buildxml does not exist!` This message indicates that Ant is installed and it is attempting to build a **build.xml** file, but the latter does not exist in the current directory. If you run Ant from a directory which includes a **build.xml** file, Ant executes the **build.xml** file instead, with whatever build options are defined in it.

To build a MapForce-generated Java project with Ant:

1. Open a command prompt and navigate to the directory where the Java project was generated (note that the directory must contain the **build.xml** file).
2. At the command prompt, enter `ant`. This will compile and execute the Java code according to the options defined in the **build.xml** file.



To generate a JAR file with Ant:

- At the command prompt, enter `ant jar`.

If you need help with Ant command syntax and options, enter `ant -help` at the command line.

16.5.5 Example: Build a Java application with Eclipse and Ant

This example walks you through the steps required to generate a Java application with MapForce, and compile it outside of MapForce using the Eclipse Integrated Development Environment (IDE) and Apache Ant. After completing this example, you will have created and compiled a complete Java application which executes one of the mapping samples available by default in MapForce.

If you can already compile successfully other Java applications with Eclipse and Ant, there are no special requirements to run this example. Otherwise, note the following prerequisites:

- The Java Development Kit (JDK), Eclipse (https://www.eclipse.org/), and Ant (http://ant.apache.org/) must be installed on your system. Eclipse typically includes support for building projects with Ant (see also Building the project with Ant).
- The Windows PATH environment variable must include the path to the JDK's **bin** directory (for example, **C:\Java\jdk1.7.0\bin**). This is a basic requirement for developing applications for the Java platform. For instructions, see http://docs.oracle.com/javase/tutorial/essential/environment/paths.html.

This example uses the following configuration:

- JDK 1.7
- Eclipse IDE for Java Developers (Luna Service Release 4.4.1)
- Ant 1.9.2, which is already integrated into the above-mentioned edition of Eclipse; therefore, it was not installed and configured separately.

The example is organized into the following sub-tasks:

- Step 1: Generate Java Code
- Step 2: Import the Project into Eclipse
- Step 3: Run the Project as GUI Application
- Step 4: Run the Project as Console Application
- Step 5: Build the JAR file with Ant

16.5.5.1 Step 1: Generate Java code

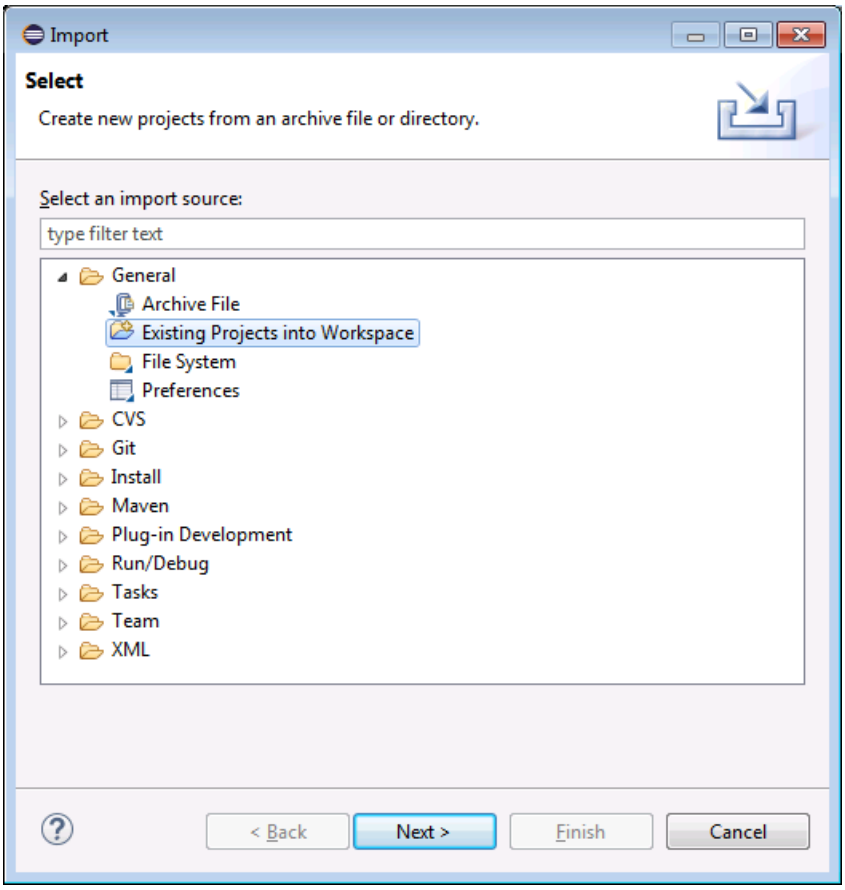
To generate the Java code in MapForce:

1. On the **File** menu, click **Open**, and browse for the **CompletePO.mfd** mapping available in the <Documents>\Altova\MapForce2017\MapForceExamples\ directory.
2. On the **Output** menu, click **Java**. This changes the transformation language to Java.
3. On the **File** menu, click **Generate code in | Java**. When prompted, browse for the directory to which the Java project must be saved. For convenience, you may choose to save the project to **C:\workspace\CompletePO** (where **C:\workspace** is your default Eclipse workspace).

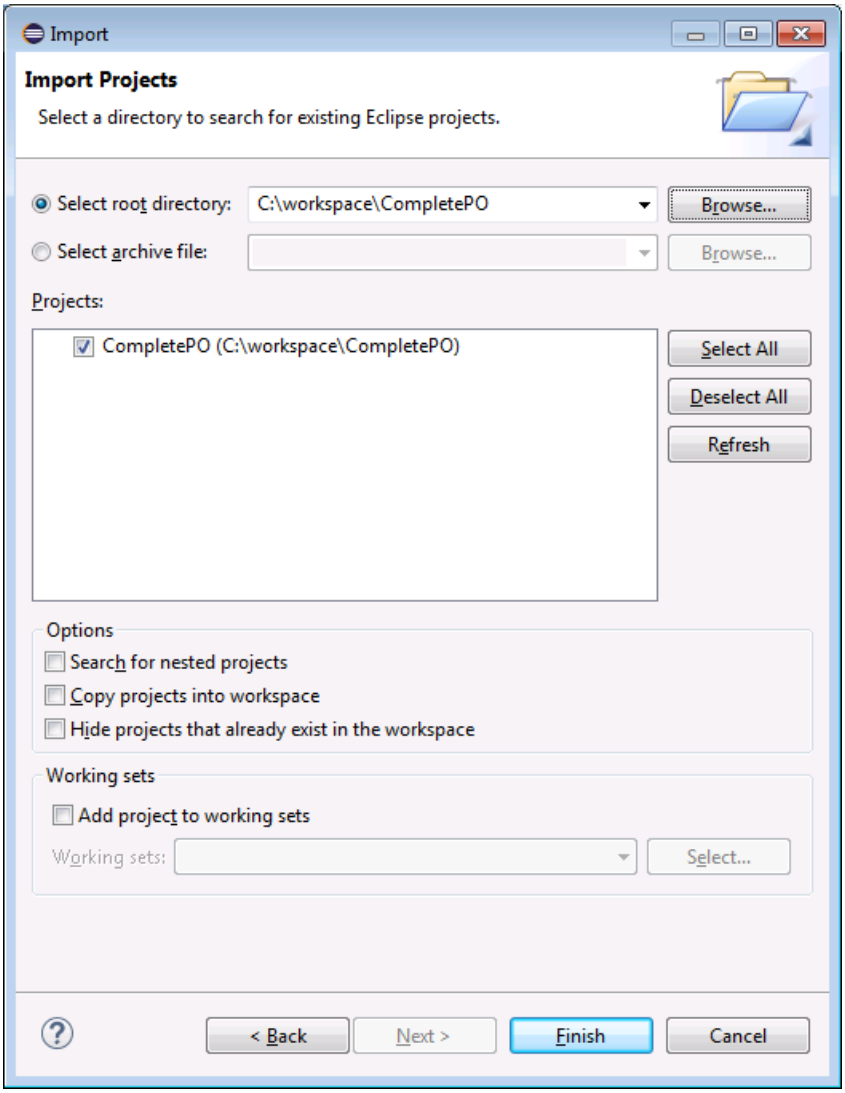
16.5.5.2 Step 2: Import the project into Eclipse

To import the project into Eclipse:

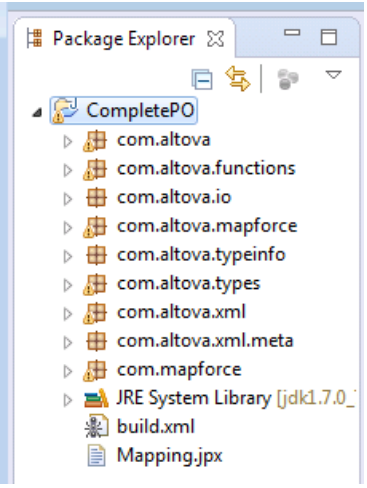
1. If you haven't done so already, run Eclipse and switch to the default Java perspective using the menu command **Window | Open Perspective**.
2. On the **File** menu, click **Import**, and then select **Existing Projects into Workspace**.



3. Click **Next**.



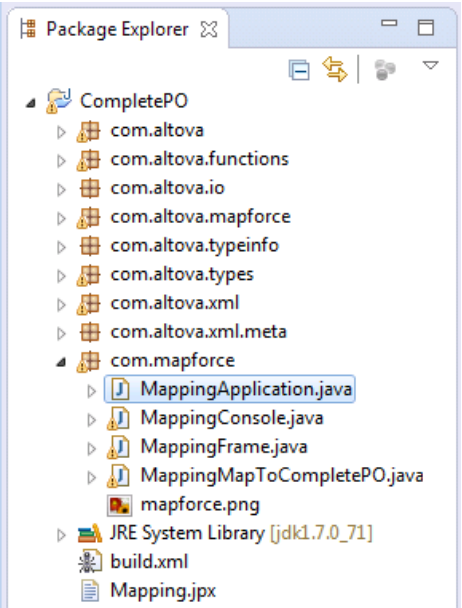
4. Browse for the folder where you have previously saved the generated code, and then click **Finish**. The Java project created by MapForce is now available in the Package Explorer view. If you cannot see the Package Explorer view, display it using the menu command **Window | Show View | Package Explorer**.



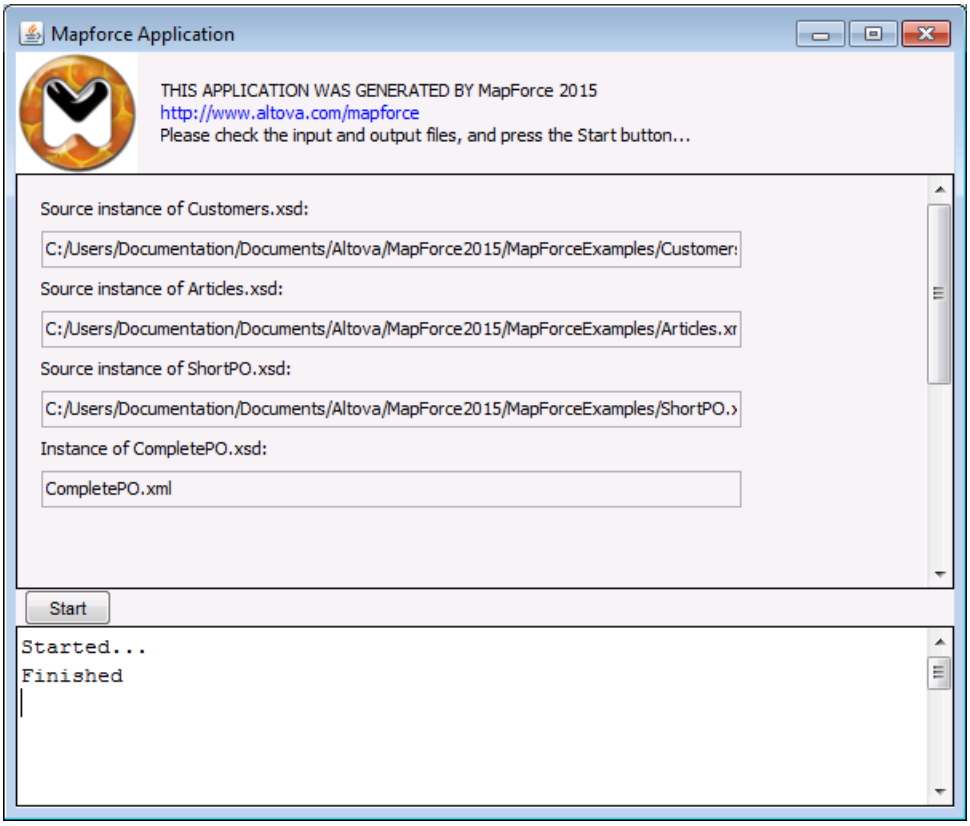
16.5.5.3 Step 3: Run the project as dialog application

To run the Java project as a GUI application:

1. In the Package Explorer view of Eclipse, click the **MappingApplication.java** file available in the **com.mapforce** package.



2. On the **Run** menu, click **Run As | Java application**.
3. On the MapForce application window, click **Start** to execute the mapping.

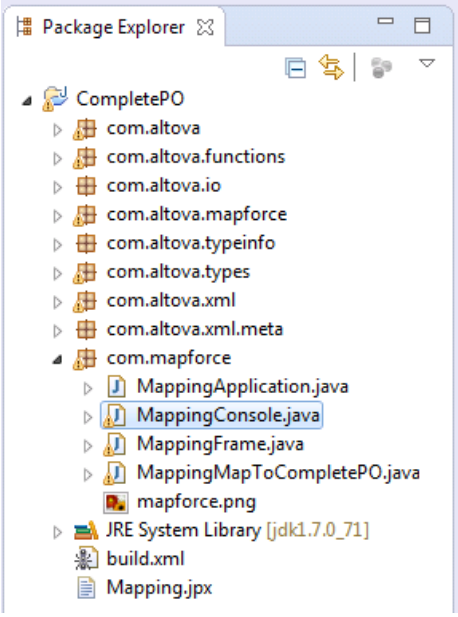


If Eclipse encounters system configuration or run-time errors, you will be prompted. Otherwise, the Java application executes the mapping transformation and generates the **CompletePO.xml** at the output path (in this example: **C:\workspace\CompletePO\CompletePO.xml**).

16.5.5.4 Step 4: Run the project as console application

To run the Java project as a console application:

1. In the Package Explorer view of Eclipse, click the **MappingConsole.java** file available in the **com.mapforce** package.



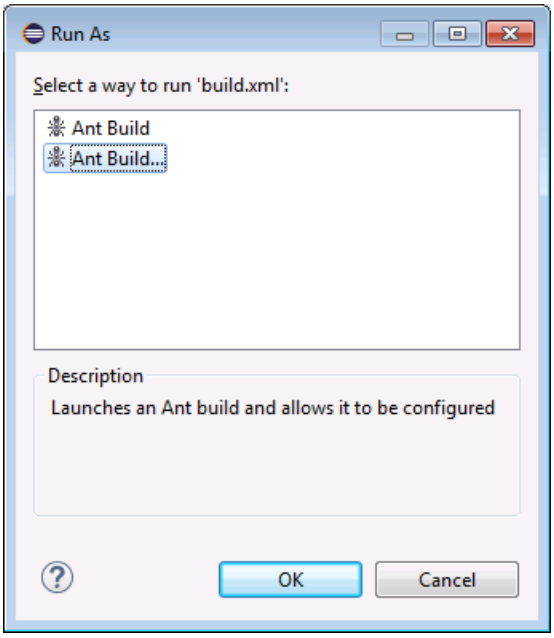
2. On the **Run** menu, click **Run As | Java application**.

If Eclipse detects system configuration or run-time errors, you will be prompted. Otherwise, the Java application executes the mapping transformation and generates the **CompletePO.xml** at the output path (in this example: **C:\workspace\CompletePO\CompletePO.xml**).

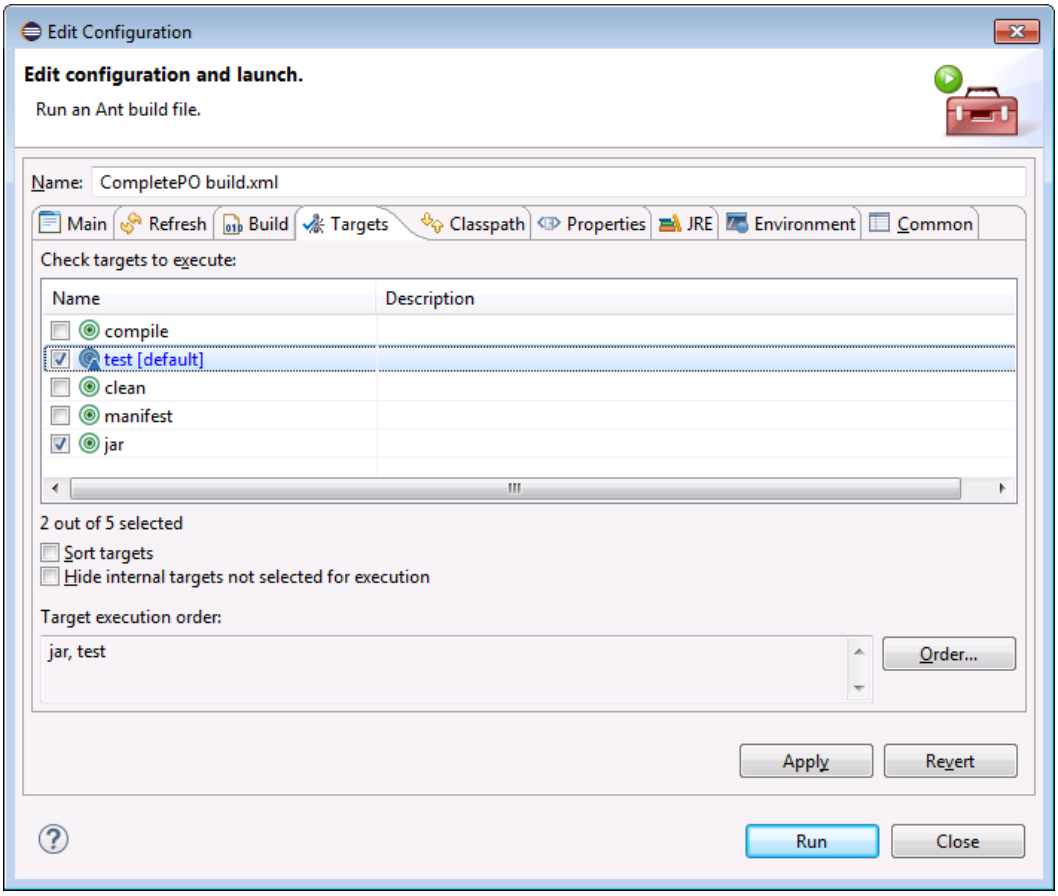
16.5.5.5 Step 5: Build the JAR file with Ant

To build the .jar file with Ant:

1. In the Package Explorer view of Eclipse, click the **build.xml** file available directly in the project root.
2. On the Run menu, click Run.



3. In the Run As dialog box, two possible options to run the Ant build file are displayed. If you choose the first option, Eclipse launches the Ant build with the default settings. If you choose the second option, you can change the settings of the Ant build before launching it. Select the second option.
4. Click to enable the targets that you wish to include in the Ant build. In this particular example, the goal is to generate a .jar file from the mapping project. Therefore, select **jar** as target, in addition to the default **test** target.



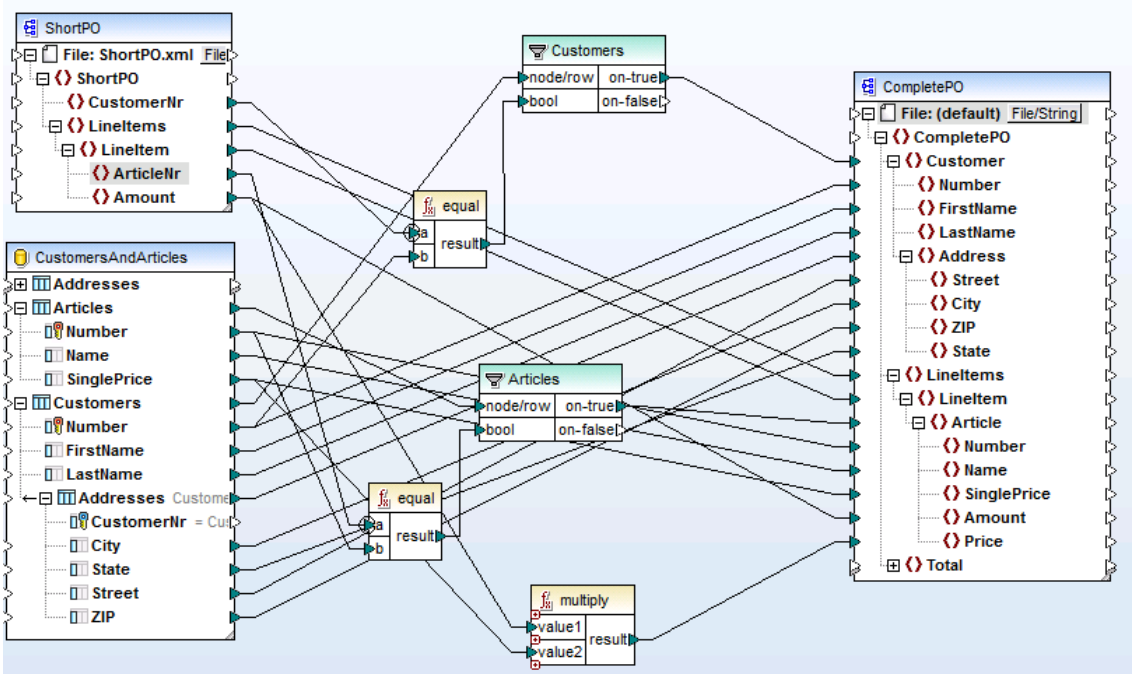
5. Click **Run**. Eclipse executes the Ant build file and displays the result in the Console view.

16.6 Integrating MapForce- Generated Code

MapForce-generated code can be integrated, or adapted to your specific application, even though the result of code generation is a complete and fully-functioning application. Some typical scenarios where you might want to change the generated code are as follows:

- Define custom source or target files for the mapping application
- Add custom error handling code
- In C# or Java generated code, you can also change the data type of the mapping input programmatically (for example, from string to stream).

This section provides instructions on how to achieve these goals, based on the **DB_CompletePO.mfd** sample mapping available in the <Documents>\Altova\MapForce2017 \MapForceExamples\ directory.



DB_CompletePO.mfd mapping sample in MapForce

As illustrated above, the sample mapping consists of two sources and one target:

- **ShortPO.xml** is a source XML file
- **CustomersAndArticles.mdb** is a source database
- **CompletePO.xml** is the target XML file.

In the generated code, these sources and targets will translate to two input and one output parameters supplied to the **`run`** method which executes the mapping (as described in the subsequent topics). For now, note the following basic points about code generation:

- The number of source and targets in the mapping design corresponds to the number of mapping parameters to the **`run`** method in the generated code.
- If you change the number of sources or targets of the mapping, then you will need to re-

generate the code accordingly.
- If you make changes to the generated code, and then re-generate the code at the same location, all changes will be overwritten.

If a mapping includes database components, the generated **run** method includes the database connection object at the appropriate location. For example, if the mapping uses three sources (text content, XML content and a database) to map to a single output file, MapForce generates the following **run** method:

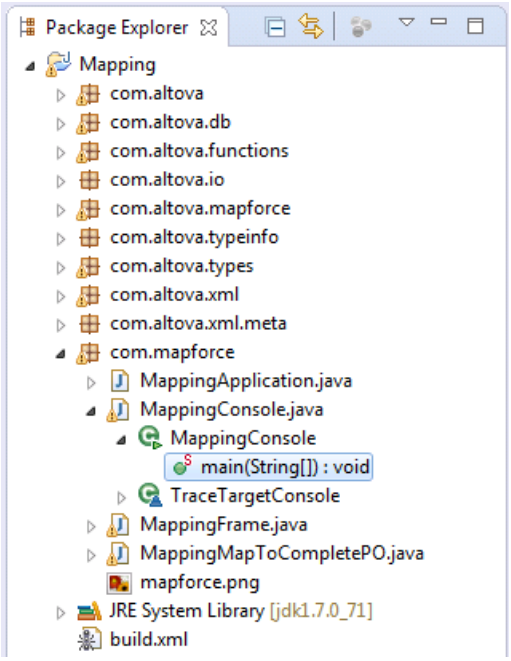
Java

```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output out1);
```

The argument order is important. As you will see in the subsequent examples, you can modify **dbConn** parameters, or use the default parameters generated by MapForce when integrating your code.

16.6.1 Java example

This example uses Eclipse as Java IDE. To begin, generate Java code from the **DB_CompletePO** sample mapping available in the <Documents>\Altova\MapForce2017\MapForceExamples\ directory, and then import the project into Eclipse.



Sample MapForce-generated Java application (Eclipse IDE)

To edit the generated Java console application, locate the `main(String[] args)` method of your generated application (see the screen shot above). If you did not change the default base package name before generating code, this method is in the `MappingConsole` class of the `com.mapforce` package. Otherwise, it is in the `MappingConsole` class of your custom defined package.

To edit the generated Java dialog application, locate the place in the code where the `run` method is invoked from your generated application. If you did not change the default base package name

before generating code, the **run** method is invoked from the class called `MappingFrame.java` of the `com.mapforce` package.

The following code sample illustrates an extract from the `main` method in the generated Java console application. The mapping sources and targets are highlighted in yellow and are defined as parameters to the run method. Since this mapping uses a database connection, the corresponding parameter has a special structure. Namely, the connection consists of the connection string (in this case, `jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=CustomersAndArticles.mdb`), as well as two empty arguments intended for the **Username** and **Password** (in clear text) for those databases where this data is necessary.

Note that the file paths in the code below have been changed from absolute to relative.

```
com.altova.io.Input ShortPO2Source =
com.altova.io.StreamInput.createInput("ShortPO.xml");
com.altova.io.Output CompletePO2Target = new
com.altova.io.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.run(
                  com.altova.db.Dbs.newConnection(
                     "jdbc:odbc:;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
                     "",
                     ""),
                  ShortPO2Source,
                  CompletePO2Target);
```

To define custom mapping source or target files:

- Locate the parameters passed to the `run` method and edit them as required. In the sample above, **com.altova.db.Dbs.newConnection** and **ShortPO2Source** is the mapping input and **CompletePO2Target** is the mapping output.

To add extra error handling code:

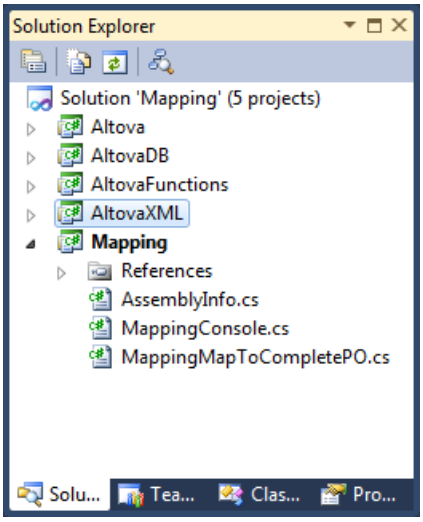
- Edit the code below the **catch** (Exception e) code (in case of a Java console application)
- Edit the code below the **catch** (Exception ex) code (in case of a Java dialog application)

For instructions on how to change the data type of parameters supplied as mapping input/output, see Changing the data type of the mapping input/output (C#, Java).

16.6.2 C# example

This example uses the Visual Studio 2010 IDE. To begin, generate C# code from the **DB_CompletePO** sample mapping available in the <Documents>\Altova\MapForce2017 \MapForceExamples\ directory, and then open the solution in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.



Sample C# solution generated with MapForce

Open the MappingConsole.cs file, and locate the `main(String[] args)` method. The following code sample illustrates an extract from the `main` method. The mapping sources and targets are highlighted in yellow and are defined as parameters to the **Run** method. Since this mapping reads data from a database, there is also an input parameter which is a database connection string. If necessary, you can modify the connection string of the database.

Note that the file paths in the code below have been changed from absolute to relative.

```
AltovaXD.Input ShortPO2Source = AltovaXD.StreamInput.createInput("ShortPO.xml");
AltovaXD.Output CompletePO2Target = new AltovaXD.FileOutput("CompletePO.xml");

MappingMapToCompletePO ObjectRun(
                                  "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=CustomersAndArticles.mdb;",
                                  ShortPO2Source,
                                  CompletePO2Target);
```

To define custom mapping source or target files:

- Locate the parameters passed to the **Run** method and edit them as required. In the sample above, the mapping input is a connection string to the CustomersAndArticles.mdb and **ShortPO2Source** . The mapping output is **CompletePO2Target** .

To add extra error handling code:

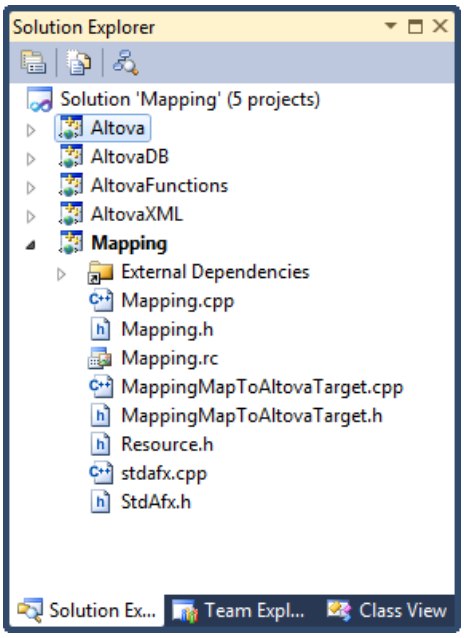
- Edit the code below the **catch** (Exception e) code

For instructions on how to change the data type of parameters supplied as mapping input/output, see Changing the data type of the mapping input/output (C#, Java).

16.6.3 C++ example

This example uses the Visual Studio 2010 IDE. To begin, generate C++ code from the **DB_CompletePO** sample mapping available in the <Documents>\Altova\MapForce2017 \MapForceExamples\ directory, and then open the solution in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.



Sample C++ solution generated with MapForce

Open the Mapping.cpp file, and locate the `_tmain` method. The following code sample illustrates an extract from this method. The mapping sources and targets are defined as parameters to the **Run** method. Since this mapping reads data from a database, there is also an input parameter which is a database connection string. If necessary, you can modify the connection string of the database.

Note that the file paths in the code below have been changed from absolute to relative.

```
MappingMapToCompletePO MappingMapToCompletePOObject;
                       MappingMapToCompletePOObjectRun(
                                  _T("Provider=MicrosoftJetOLEDB.4.0;Data
Source=CustomersAndArticles.mdb;"),
                                  _T("ShortPO.xml"),
                                  _T("CompletePO.xml"));
```

To define custom mapping source or target files:

- Locate the parameters passed to the **Run** method and edit them as required. In the code

sample above, the mapping input is a connection string to the **CustomersAndArticles.mdb** database and `_T("ShortPO.xml")` . The mapping output is `_T("CompletePO.xml")` .

To add extra error handling code:

- Edit the code below the `catch (CAltovaException& e)` code.

16.6.4 Changing the data type of the mapping input/output (C#, Java)

This topic provides details on the object types you can use programmatically, if you intend to run MapForce mappings from a custom Java or C# application.

You can use several input and output objects (such as files, strings, DOM documents, and others) as parameters to the **run** method. The **run** method is the most important function of generated mapping classes. It has one parameter for each static source or input component in the mapping, and a final parameter for the output component. Components that process multiple files do not appear as parameters to the **run** method, because in this case the file names are processed dynamically inside the mapping.

The objects that you can provide as parameters to the **run** method are available in the **com.altova.io** package (Java) and Altova.IO namespace (C#). The base classes of the generated input and output objects are as follows:

C#

```
Altova.IO.Input
Altova.IO.Output
```

Java

```
com.altova.io.Input
com.altova.io.Output
```

The object types supported as input/output parameters to the **run** method, including their applicable input/output file formats, are listed in the following table.

Object Type	XML	Microsoft Excel*	EDI* **	FlexText*	CSV	Fixed-length files
Files	Y	Y	Y	Y	Y	Y
Binary stream objects	Y	Y	Y	Y	Y	Y
Strings	Y	–	Y	Y	Y	Y
I/O Reader/Writer (character stream objects)	Y	–	Y	Y	Y	Y
DOM documents	Y	–	–	–	–	–

***** Formats supported only in MapForce Enterprise Edition
****** Includes X12 and HL7

Files

File objects (identified in the code file names) have the following definition:

C#

```
Altova.IO.FileInput(string filename)
Altova.IO.FileOutput(string filename)
```

Java

```
com.altova.io.FileInput(String filename)
com.altova.io.FileOutput(String filename)
```

Binary stream objects

Binary stream objects in the generated code represent an alternative way to working with physical files; there are no advantages as far as memory use is concerned. Binary stream objects have the following definition:

C#

```
Altova.IO.StreamInput(System.IO.Stream stream)
Altova.IO.StreamOutput(System.IO.Stream stream)
```

Java

```
com.altova.io.StreamInput(java.io.InputStream stream)
com.altova.io.StreamOutput(java.io.OutputStream stream)
```

Notes:

- Binary stream objects are expected to be opened and ready-to-use before calling the **run** method.
- By default, the **run** method closes the stream when finished. To prevent this behaviour, insert the following code before calling the **run** method:

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);  // Java
```

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;   // C#
```

Strings

String objects have the following definition:

C#

```
Altova.IO.StringInput(string content)
Altova.IO.StringOutput(StringBuilder content)
```

Java

```
com.altova.io.StringInput(String xmlcontent)
com.altova.io.StringOutput()
```

In Java, **StringOutput** does not take an argument. Content can be accessed with:

```
// mapping from String to (another) String
String MyText = "<here>is some XML text</here>";

Input input = new StringInput(MyText);
Output output = new StringOutput();

MappingMapToMyText.run(input, output);

String myTargetData = output.getString().toString();
```

The **getString()** method returns a **StringBuffer**, hence the need for **toString()**.

In C#, **StringOutput** takes an argument (**StringBuilder**) which you need to provide beforehand. The **StringBuilder** may already contain data, so the mapping output is appended to it.

Excel sources/targets cannot map to or from strings.

I/O Reader/Writer (character stream objects)

Character stream objects have the following definition:

C#

```
Altova.IO.ReaderInput(System.IO.TextReader reader)
Altova.IO.WriterOutput(System.IO.TextWriter writer)
```

Java

```
com.altova.io.ReaderInput(java.io.Reader reader)
com.altova.io.WriterOutput(java.io.Writer writer)
```

Notes:

- Character stream objects are expected to be opened and ready-to-use before calling the **run** method.
- Excel sources/targets cannot be read from, or written to, character streams.
- By default, the **run** method closes the stream when finished. To prevent this behaviour, insert the following code before calling the **run** method:

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);  // Java
```

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;   // C#
```

DOM documents

DOM documents have the following definition:

C#

```
Altova.IO.DocumentInput(System.Xml.XmlDocument document)
Altova.IO.DocumentOutput(System.Xml.XmlDocument document)
```

Java

```
com.altova.io.DocumentInput(org.w3c.dom.Document document)
com.altova.io.DocumentOutput(org.w3c.dom.Document document)
```

Notes:

- The document passed to the **DocumentOutput** constructor as target must be empty.
- After calling **run**, the DOM Document generated by the constructor of **DocumentOutput** already contains mapped data so "save to document" is not necessary. After mapping, you can manipulate the document as necessary.
- Only XML content can be mapped to DOM documents.

Example

Let's assume you want to integrate the code generated by MapForce into your Java application. Your MapForce mapping consists of two source XML files and a target text file. When you generate the MapForce code, the **run** function looks as follows:

```
void run(Input in1, Input in2, Output out1);
```

Let's also assume that your application requires that you map data from a local file and binary stream into a character stream. Since data is supplied from other sources, your application must declare the sources and targets as:

```
String filename; // Declare the source of the first input
Java.io.InputStream stream; // Declare the source of the second input
Java.io.Writer writer; // Declare the output as character stream
```

The following wrappers must be constructed for the MapForce- generated **run** function:

```
// com.altova.io is considered imported here:
Input input1 = new FileInput(filename);
Input input2 = new StreamInput(stream);
Output output1 = new WriterOutput(writer);
```

Now you can call the MapForce generated run function:

```
MappingMapToSomething.run(input1, input2, output1);
```

The **C#** behavior is almost identical, except that **run** is called **Run**, and the .NET stream and reader/ writer classes are named differently.

Using the same technique, you can also use other input and output types, such as strings or DOM documents.

16.7 Generating Code from XML Schemas or DTDs

When you generate code from a mapping, MapForce generates a complete application that executes all steps of the mapping automatically. Optionally, you can generate libraries for all the XML schemas used in the mapping. These allow your code to easily create or read XML instances that are used or created by the mapping code.

To generate libraries for all the XML schemas used in the mapping, select the **Generate Wrapper Classes** check box in the Options dialog (see Code Generator Options). Next time when you generate code, MapForce will create not only the mapping application, but also wrapper classes for all schemas used in the mapping, as follows:

C++ or C#	Java	Purpose
Altova	`com.altova`	Base library containing common runtime support, identical for every schema.
AltovaXML	`com.altova.xml`	Base library containing runtime support for XML, identical for every schema.
YourSchema	*`com.YourSchema`*	A library containing declarations generated from the input schema, named as the schema file or DTD. This library is a DOM (W3C Document Object Model) wrapper that allows you to read, modify and create XML documents easily and safely. All data is held inside the DOM, and there are methods for extracting data from the DOM, and to update and create data into the DOM. The generated C++ code supports either Microsoft MSXML or Apache Xerces 2.6 or higher, depending on code generation settings. The syntax for using the generated code is identical for both DOM implementations. The generated C# code uses the .NET standard System.XML library as the underlying DOM implementation. The generated Java code uses JAXP (Java API for XML Processing) as the underlying DOM interface.

While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

In addition to the base libraries listed above, some supporting libraries are also generated. The supporting libraries are used by the Altova base libraries and are not meant for custom integrations, since they are subject to change.

Name generation and namespaces

MapForce generates classes corresponding to all declared elements or complex types which redefine any complex type in your XML Schema, preserving the class derivation as defined by extensions of complex types in your XML Schema. In the case of complex schemas which import schema components from multiple namespaces, MapForce preserves this information by generating the appropriate C# or C++ namespaces or Java packages.

Generally, the code generator tries to preserve the names for generated namespaces, classes and members from the original XML Schema. Characters that are not valid in identifiers in the target language are replaced by a "_". Names that would collide with other names or reserved words are made unique by appending a number. Name generation can be influenced by changing the default settings in the SPL (Spy Programming Language) template.

The namespaces from the XML Schema are converted to packages in Java or namespaces in C# or C++ code, using the namespace prefix from the schema as code namespace. The complete library is enclosed in a package or namespace derived from the schema file name, so you can use multiple generated libraries in one program without name conflicts.

Data Types

XML Schema has a more elaborate data type model than Java, C# or C++. Code Generator converts the built-in XML Schema types to language-specific primitive types, or to classes delivered with the Altova library. Complex types and derived types defined in the schema are converted to classes in the generated library. Enumeration facets from simple types are converted to symbolic constants.

The mapping of simple types can be configured in the SPL template, see SPL (Spy Programming Language).

If your XML instance files use schema types related to time and duration, these are converted to Altova native classes in the generated code. For information about the Altova library classes, see:

- Reference to Generated Classes (C++)
- Reference to Generated Classes (C#)
- Reference to Generated Classes (Java)

For information about type conversion and other details applicable to each language, see:

- About Schema Wrapper Libraries (C++)
- About Schema Wrapper Libraries (C#)
- About Schema Wrapper Libraries (Java)

Memory management

A DOM tree is comprised of nodes, which are always owned by a specific DOM document - even if the node is not currently part of the document's content. All generated classes are references to the DOM nodes they represent, not values. This means that assigning an instance of a generated class does not copy the value, it only creates an additional reference to the same data.

XML Schema support

The following XML Schema constructs are translated into code:

a) XML namespaces

b) Simple types:

- Built-in XML schema types
- Simple types derived by extension
- Simple types derived by restriction
- Facets
- Enumerations
- Patterns

c) Complex types:

- Built-in anyType node
- User-defined complex types
- Derived by extension: Mapped to derived classes
- Derived by restriction
- Complex content
- Simple content
- Mixed content

The following advanced XML Schema features are not supported (or not fully supported) in generated wrapper classes:

- Wildcards: `xs:any` and `xs:anyAttribute`
- Content models (sequence, choice, all). Top-level compositor is available in SPL (Spy Programming Language), but is not enforced by generated classes.
- Default and fixed values for attributes. These are available in SPL (Spy Programming Language), but are not set or enforced by generated classes.
- The attributes `xsi:type`, abstract types. When you need to write the xsi:type attribute, use the `SetXsiType()` method of the generated classes.
- Union types: not all combinations are supported.
- Substitution groups are partially supported (resolved like "choice").
- Attribute `nillable="true"` and `xsi:nil`
- Uniqueness constraints
- Identity constraints (`key` and `keyref`)

16.7.1 About Schema Wrapper Libraries (C++)

Character Types

The generated C++ code can be compiled with or without Unicode support. Depending on this setting, the types `string_type` and `tstring` will both be defined as `std::string` or `std::wstring`, consisting of narrow or wide characters. To use Unicode characters in your XML file that are not representable with the current 8-bit character set, Unicode support must be enabled. Pay special attention to the `_T()` macros. This macro ensures that string constants are stored correctly, whether you're compiling for Unicode or non-Unicode programs.

Data Types

The default mapping of XML Schema types to C++ data types is:

XML Schema	C++	Remarks
xs:string	string_type	string_type is defined as std::string or std:wstring
xs:boolean	bool	
xs:decimal	double	C++ does not have a decimal type, so double is used.
xs:float, xs:double	double	
xs:integer	__int64	xs:integer has unlimited range, mapped to _int64 for efficiency reasons.
xs:nonNegativeInteger	unsigned __int64	see above
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 xs:base64Binary	std::vector<unsigned char>	Encoding and decoding of binary data is done automatically.
xs:anySimpleType	string_type	

All XML Schema types not contained in this list are derived types, and mapped to the same C++ type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

`altova::meta::SimpleType`
`altova::meta::ComplexType`

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "CDoc" below) is generated. It contains all possible root elements as members, and various

other methods. For more information about the class, see `[YourSchema]::[CDoc]` .

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

`[YourSchema]::MemberAttribute`
`[YourSchema]::MemberElement`

Note: The actual class names depend on the name of the schema attribute or element.

See also Example: Using the Schema Wrapper Libraries.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace `altova`:

Class	Base Class	Description
`Error`	`std::logic_error`	Internal program logic error (independent of input data).
`Exception`	`std::runtime_error`	Base class for runtime errors.
`InvalidArgumentsException`	`Exception`	A method was called with invalid argument values.
`ConversionException`	`Exception`	Exception thrown when a type conversion fails.
`StringParseException`	`ConversionException`	A value in the lexical space cannot be converted to value space.
`ValueNotRepresentableException`	`ConversionException`	A value in the value space cannot be converted to lexical space.
`OutOfRangeException`	`ConversionException`	A source value cannot be represented in target domain.
`InvalidOperationException`	`Exception`	An operation was attempted that is not valid in the given context.
`DataSourceUnavailableException`	`Exception`	A problem occurred while loading an XML instance.
`DataTargetUnavailableException`	`Exception`	A problem occurred while saving an XML instance.

All exception classes contain a message text and a pointer to a possible inner exception.

Method	Purpose
`string_type message()`	Returns a textual description of the exception.

(C) 2017 Altova GmbH

Method	Purpose
`std::exception inner()`	Returns the exception that caused this exception, if available, or NULL.

Accessing schema information

The generated library allows accessing static schema information via the following classes. All methods are declared as `const`. The methods that return one of the metadata classes return a NULL object if the respective property does not exist.

`altova::meta::Attribute`
`altova::meta::ComplexType`
`altova::meta::Element`
`altova::meta::SimpleType`

16.7.2 About Schema Wrapper Libraries (C#)

The default mapping of XML Schema types to C# data types is as follows.

XML Schema	C#	Remarks
`xs:string`	`string`	
`xs:boolean`	`bool`	
`xs:decimal`	`decimal`	xs:decimal has unlimited range and precision, mapped to decimal for efficiency reasons.
`xs:float, xs:double`	`double`	
`xs:long`	`long`	
`xs:unsignedLong`	`ulong`	
`xs:int`	`int`	
`xs:unsignedInt`	`uint`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`Altova.Types.DateTime`	
`xs:duration`	`Altova.Types.Duration`	
`xs:hexBinary and xs:base64Binary`	`byte[]`	Encoding and decoding of binary data is done automatically.
`xs:anySimpleType`	`string`	

All XML Schema types not contained in this list are derived types, and mapped to the same C# type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

`Altova.Xml.Meta.SimpleType`
`Altova.Xml.Meta.ComplexType`

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "Doc" below) is generated. It contains all possible root elements as members, and various other methods. For more information about the class, see `[YourSchema].[Doc]`.

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

`[YourSchemaType].MemberAttribute`
`[YourSchemaType].MemberElement`

Note: The actual class names depend on the name of the schema attribute or element.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace Altova:

Class	Base Class	Description
`ConversionException`	`Exception`	Exception thrown when a type conversion fails
`StringParseException`	`ConversionException`	A value in the lexical space cannot be converted to value space.
`DataSourceUnavailableException`	`System.Exception`	A problem occurred while loading an XML instance.
`DataTargetUnavailableException`	`System.Exception`	A problem occurred while saving an XML instance.

In addition, the following .NET exceptions are commonly used:

Class	Description
`System.Exception`	Base class for runtime errors
`System.ArgumentException`	A method was called with invalid argument values, or

Class	Description
	a type conversion failed.
`System.FormatException`	A value in the lexical space cannot be converted to value space.
`System.InvalidCastException`	A value cannot be converted to another type.
`System.OverflowException`	A source value cannot be represented in target domain.

Accessing schema information

The generated library allows accessing static schema information via the following classes:

`Altova.Xml.Meta.Attribute`
`Altova.Xml.Meta.ComplexType`
`Altova.Xml.Meta.Element`
`Altova.Xml.Meta.SimpleType`

The properties that return one of the metadata classes return null if the respective property does not exist.

16.7.3 About Schema Wrapper Libraries (Java)

The default mapping of XML Schema types to Java data types is as follows:

XML Schema	Java	Remarks
`xs:string`	`String`	
`xs:boolean`	`boolean`	
`xs:decimal`	`java.math.BigDecimal`	
`xs:float, xs:double`	`double`	
`xs:integer`	`java.math.BigInteger`	
`xs:long`	`long`	
`xs:unsignedLong`	`java.math.BigInteger`	Java does not have unsigned types.
`xs:int`	`int`	
`xs:unsignedInt`	`long`	Java does not have unsigned types.
`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[]`	Encoding and decoding of binary data is done automatically.

XML Schema	Java	Remarks
xs:anySimpleType	string	

All XML Schema types not contained in this list are derived types, and mapped to the same Java type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

com.altova.xml.meta.SimpleType
com.altova.xml.meta.ComplexType

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "Doc" below) is generated. It contains all possible root elements as members, and various other methods. For more information about the class, see com.[YourSchema].[Doc].

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

com.[YourSchema].[YourSchemaType].MemberAttribute
com.[YourSchema].[YourSchemaType].MemberElement

Note: The actual class names depend on the name of the schema attribute or element.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace com.altova:

Class	Base Class	Description
SourceInstanceUnvailableException	Exception	A problem occurred while loading an XML instance.
TargetInstanceUnavailableException	Exception	A problem occurred while saving an XML instance.

In addition, the following Java exceptions are commonly used:

Class	Description
`java.lang.Error`	Internal program logic error (independent of input data)
`java.lang.Exception`	Base class for runtime errors
`java.lang.IllegalArgumentsException`	A method was called with invalid argument values, or a type conversion failed.
`java.lang.ArithmeticException`	Exception thrown when a numeric type conversion fails.

Accessing schema information

The generated library allows accessing static schema information via the following classes:

`com.altova.xml.meta.Attribute`
`com.altova.xml.meta.ComplexType`
`com.altova.xml.meta.Element`
`com.altova.xml.meta.SimpleType`

The properties that return one of the metadata classes return null if the respective property does not exist.

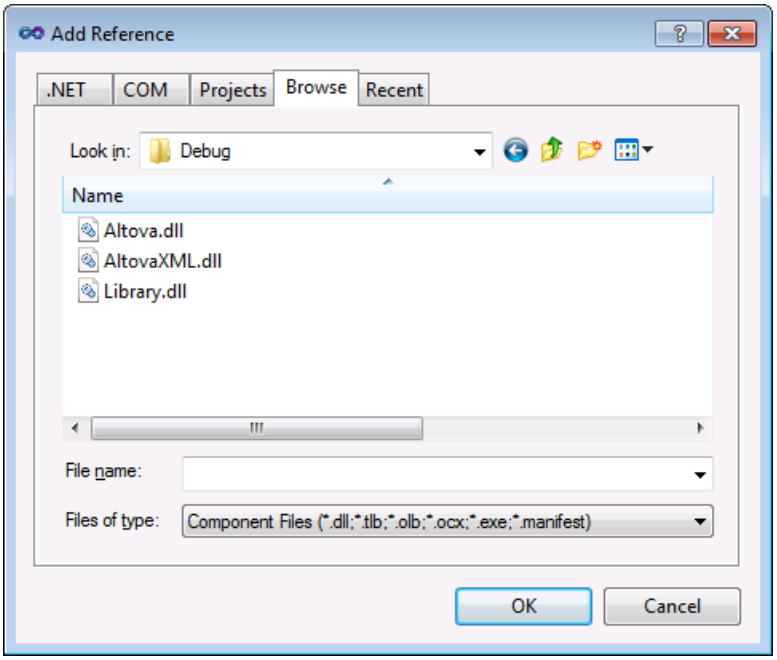
16.7.4 Integrating Schema Wrapper Libraries

To use the Altova libraries in your custom project, refer to the libraries from your project (or include them into your project), as shown below for each language.

C#

To integrate the Altova libraries into an existing C# project:

1. After MapForce generates code from a schema (for example, **YourSchema.xsd**), build the generated **YourSchema.sln** solution in Visual Studio. This solution is in a project folder with the same name as the schema.
2. Right-click your existing project in Visual Studio, and select **Add Reference**.
3. On the Browse tab, browse for the following libraries: **Altova.dll**, **AltovaXML.dll**, and **YourSchema.dll** located in the output directory of the generated projects (for example, **bin \Debug**).



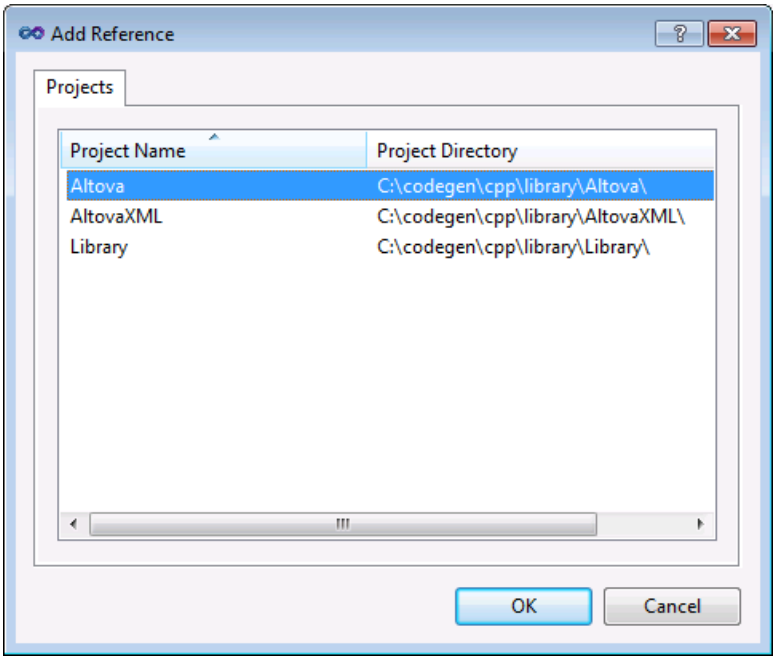
C++

The easiest way to integrate the libraries into an existing C++ project is to add the generated project files to your solution. For example, let's assume that you generated code from a schema called **Library.xsd** and selected **c:\codegen\cpp\library** as target directory. The generated libraries in this case are available at:

- c:\codegen\cpp\library\Altova.vcxproj
- c:\codegen\cpp\library\AltovaXML\AltovaXML.vcxproj
- c:\codegen\cpp\library\Library.vcxproj

First, open the generated **c:\codegen\cpp\library\Library.sln** solution and build it in Visual Studio.

Next, open your existing Visual Studio solution (in Visual Studio 2010, in this example), right-click it, select **Add | Existing Project**, and add the project files listed above, one by one. Be patient while Visual Studio parses the files. Next, right-click your project and select **Properties**. In the Property Pages dialog box, select **Common Properties | Framework and References**, and then click **Add New Reference**. Next, select and add each of the following projects: *Altova*, *AltovaXML*, and *Library*.



See also the MSDN documentation for using functionality from a custom library, as applicable to your version of Visual Studio, for example:

- If you chose to generate static libraries, see https://msdn.microsoft.com/en-us/library/ms235627(v=vs.100).aspx
- If you chose to generate dynamic libraries, see https://msdn.microsoft.com/en-us/library/ms235636(v=vs.100).aspx

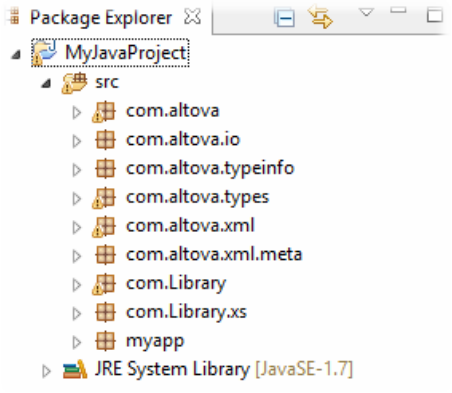
The option to generate static or dynamic libraries is available in code generation options (see Code generator options).

Java

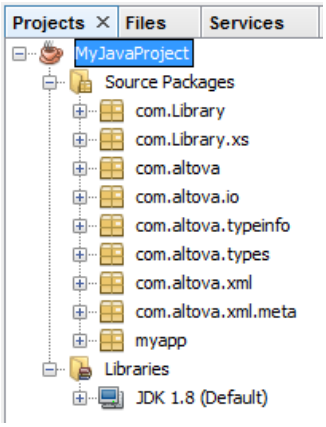
One of the ways to integrate the Altova packages into your Java project is to copy the **com** directory of the generated code to the directory which stores the source packages of your Java project (for example, **C:\Workspace\MyJavaProject\src**). For example, let's assume that you generated code in **c:\codegen\java\library**. The generated Altova classes in this case are available at **c:\codegen\java\library\com**.

After copying the libraries, refresh the project. To refresh the project in Eclipse, select it in the Package Explorer, and press **F5**. To refresh the project in NetBeans IDE 8.0, select the menu command **Source | Scan for External Changes**.

Once you perform the copy operation, the Altova packages are available in the Package Explorer (in case of Eclipse), or under "Source Packages" in the Projects pane (in case of NetBeans IDE).



Altova packages in Eclipse 4.4



Altova packages in NetBeans IDE 8.0.2

16.7.5 Example: Using the Schema Wrapper Libraries

This example illustrates how to use the generated schema wrapper libraries in order to write or read programmatically XML documents conformant to the schema. Before using the sample code, take some time to understand the structure of the included example schema. You will need this schema to generate the code libraries used in this example. Understanding the example schema will help you get started with the code generated from your schema and adapt it to your needs.

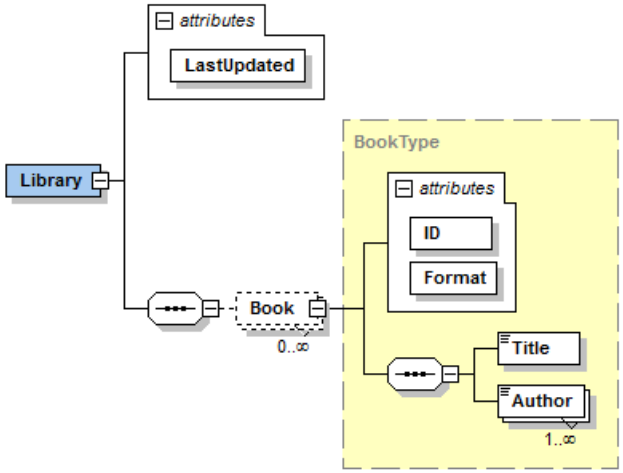
16.7.5.1 Example Schema

The schema used in this example describes a library of books. The complete definition of the schema is shown below. Save this code listing as **Library.xsd** if you want to get the same results as this example. You will need this schema to generate the code libraries used in this example.

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xs:schema xmlns="http://www.nanonull.com/LibrarySample" xmlns:xs="http://
www.w3.org/2001/XMLSchema" targetNamespace="http://www.nanonull.com/
LibrarySample" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="Library">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Book" type="BookType" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LastUpdated" type="xs:dateTime"/>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="BookType">
      <xs:sequence>
         <xs:element name="Title" type="xs:string"/>
         <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:integer" use="required"/>
      <xs:attribute name="Format" type="BookFormatType" use="required"/>
   </xs:complexType>
   <xs:complexType name="DictionaryType">
      <xs:complexContent>
         <xs:extension base="BookType">
            <xs:sequence>
               <xs:element name="FromLang" type="xs:string"/>
               <xs:element name="ToLang" type="xs:string"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="BookFormatType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Hardcover"/>
         <xs:enumeration value="Paperback"/>
         <xs:enumeration value="Audiobook"/>
         <xs:enumeration value="E-book"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

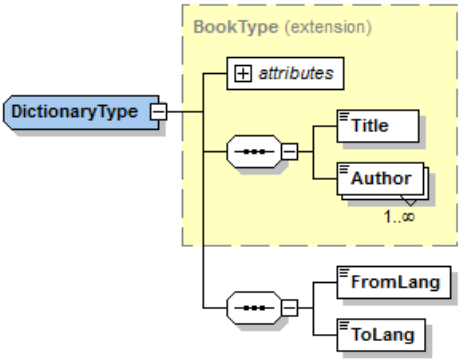
Library is a root element of a `complexType` which can be graphically represented as follows in the schema view of XMLSpy:



As shown above, the library has a **LastUpdated** attribute (defined as `xs:dateTime`), and stores a sequence of books. Each book is an `xs:complexType` and has two attributes: an **ID** (defined as `xs:integer`), and a **Format**. The format of any book can be hardcover, paperback, audiobook, or e-book. In the schema, **Format** is defined as `xs:simpleType` which uses an enumeration of the above-mentioned values.

Each book also has a **Title** element (defined as `xs:string`), as well as one or several **Author** elements (defined as `xs:string`).

The library may also contain books that are dictionaries. Dictionaries have the type `DictionaryType`, which is derived by extension from the `BookType`. In other words, a dictionary inherits all attributes and elements of a Book, plus two additional elements: **FromLang** and **ToLang**, as illustrated below.



The **FromLang** and **ToLang** elements store the source and destination language of the dictionary.

An XML instance file valid according to the schema above could therefore look as shown in the listing below (provided that it is in the same directory as the schema file):

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample" xsi:schemaLocation="http://
www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-02-
03T17:10:08.4977404">
```

```
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

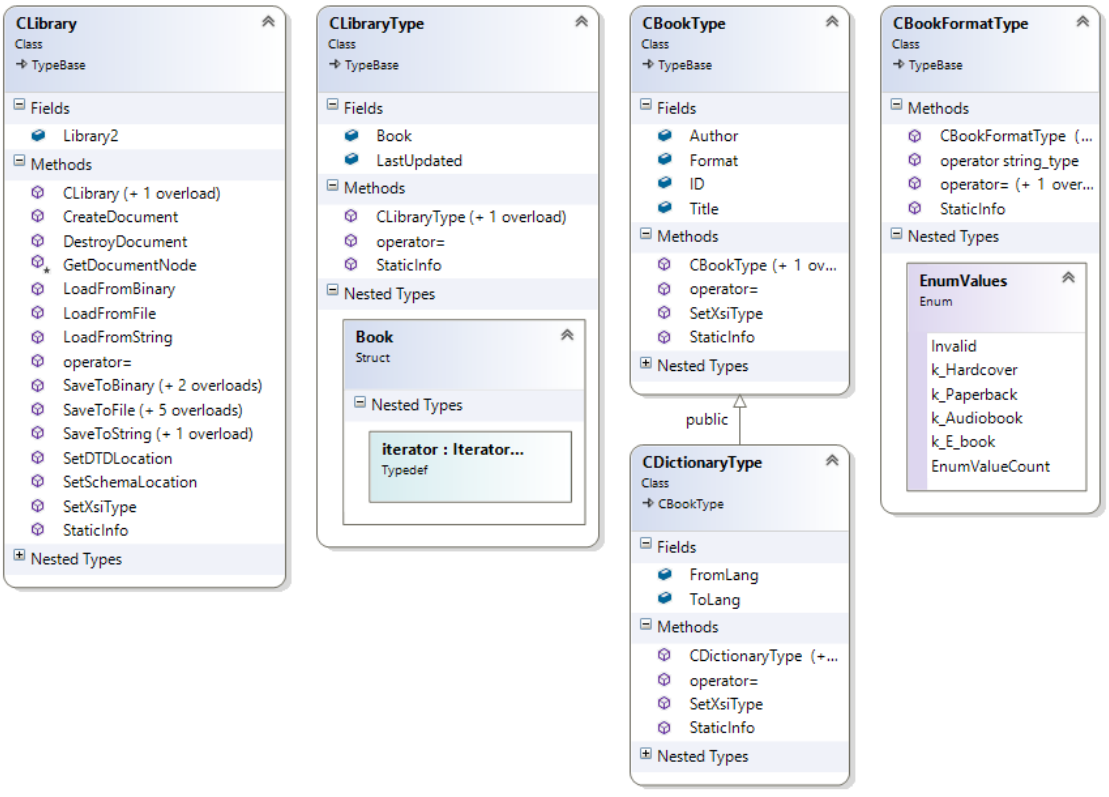
The next topics illustrate how to read from such a file programmatically, or write to such a file programmatically. To begin, generate the schema wrapper code from the schema above, using the steps described in Generating Code from XML Schemas or DTD.

16.7.5.2 Reading and Writing XML Documents (C++)

After you generate code from the Library schema (see Example Schema), a test C++ application is created, along with several supporting Altova libraries.

About the generated C++ libraries

The following diagram illustrates some of the most important classes of the generated code.



The central class of the generated code is the `CLibrary` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Library.xsd**, in this example). As shown in the diagram, this class provides methods for loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a list of all members exposed by this class, see the class reference (`[YourSchema]::[CDoc]`).

The `Library2` field of the `CLibrary` class represents the actual root of the document. The number at the end is meant to avoid a naming conflict with the class name. **Library** is an element in the XML file, so in the C++ code it has a template class as type (`MemberElement`). The template class exposes methods and properties for interacting with the **Library** element. In general, each attribute and each element of a type in the schema is typed in the generated code with the `MemberAttribute` and `MemberElement` template classes, respectively. For more information, see `[YourSchema]::MemberAttribute` and `[YourSchema]::MemberElement` class reference.

The class `CLibraryType` is generated from the schema complex type with the same name, as mentioned in About Schema Wrapper Libraries (C++). Notice that the `CLibraryType` class contains a field `Book`, and a field `LastUpdated`. According to the logic already mentioned above, these correspond to the **Book** element and **LastUpdated** attribute in the schema, and enable you to manipulate programmatically (append, remove, etc) elements and attributes in the instance XML document.

Since the **DictionaryType** is a complex type derived from **BookType** in the schema, this relationship is also reflected in the generated classes. As illustrated in the diagram, the class `CDictionaryType` inherits the `CBookType` class.

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` that is a member of the `CBookFormatType` class.

Writing an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio generated from the Library schema mentioned earlier in this example.

> While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

2. In Solution Explorer, open the **LibraryTest.cpp** file, and edit the `Example()` method as shown below.

```
#include <ctime> // required to get current time
using namespace Library;// required to work with Altova libraries

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();

        // Get current time and set the "LastUpdated" attribute using Altova classes
        time_t t = time(NULL);
        struct tm * now = localtime( & t );
        altova:DateTime dt = altova:DateTime(now->tm_year + 1900,now->tm_mon + 1,now-
>tm_mday,now->tm_hour,now->tm_min,now->tm_sec);
        lib.LastUpdated = dt;

        // Create a new <Book> and add it to the library
        CBookType book = lib.Book.append();

        // Set the "ID" attribute of the book
        book.ID = 1;

        // Set the "Format" attribute of the <Book> using an enumeration constant
        book.FormatSetEnumerationValue(CBookFormatType::k_Paperback );

        // Add the <Title> and <Author> elements, and set values
        book.Title.append() = _T("The XML Spy Handbook");
        book.Author.append() = _T("Altova");

        // Append a dictionary (book of derived type) and populate its attributes and elements
        CDictionaryType dictionary = CDictionaryType(lib.Book.append().GetNode());
        dictionary.ID = 2;
        dictionary.FormatSetEnumerationValue(CBookFormatType::k_E_book);
        dictionary.Title.append() = _T("English-German Dictionary");
        dictionary.Author.append() = _T("John Doe");
        dictionary.FromLang.append() = _T("English");
        dictionary.ToLang.append() = _T("German");

        // Since dictionary a derived type, set the xsitype attribute of the book element
        dictionary.SetXsiType();

        // Optionally, set the schema location
        libDoc.SetSchemaLocation(_T("Library.xsd"));

        // Save the XML document to a file with default encoding (UTF-8),
        // "true" causes the file to be pretty-printed.
        libDoc.SaveToFile(_T("GeneratedLibrary.xml"),true);

        // Destroy the document
        libDoc.DestroyDocument();
}
```

3. Press **F5** to start debugging. If the code was executed successfully, a **GeneratedLibrary.xml** file is created in the solution output directory.

Reading an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio.
2. Save the code below as **Library1.xml** to a directory that can be read by the program code

(for example, the same directory as **LibraryTest.sln**).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample" xsi:schemaLocation="http://
www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-02-
03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. In Solution Explorer, open the **LibraryTest.cpp** file, and edit the `Example()` method as shown below.

```
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();

	// Check whether an element exists:
	if(!lib.Book.exists())
	{
		tcout << "This library is empty." << std::endl;
		return;
	}

	// iteration: for each <Book> ...
	for(Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
	{
		// output values of ISBN attribute and (first and only) title element
		tcout << "ID:" << itBook->ID << std::endl;
		tcout << "Title:" << tstring(itBook->Title.first()) << std::endl;

		// read and compare an enumeration value
		if(itBook->Format.GetEnumerationValue() == CBookFormatType::k_Paperback)
			tcout << "This is a paperback book." << std::endl;

		// for each <Author> ...
		for(CBookType::Author::iterator itAuthor = itBook->Author.all(); itAuthor;
++itAuthor)
			tcout << "Author:" << tstring(itAuthor) << std::endl;

		// alternative: use count and index
```

```
            for (unsigned int j= 0;j< itBook->Author.count();++j)
                tcout << "Author:" << tstring(itBook->Author[j]) << std::endl;
        }

        // Destroy the document
        libDoc.DestroyDocument();
}
```

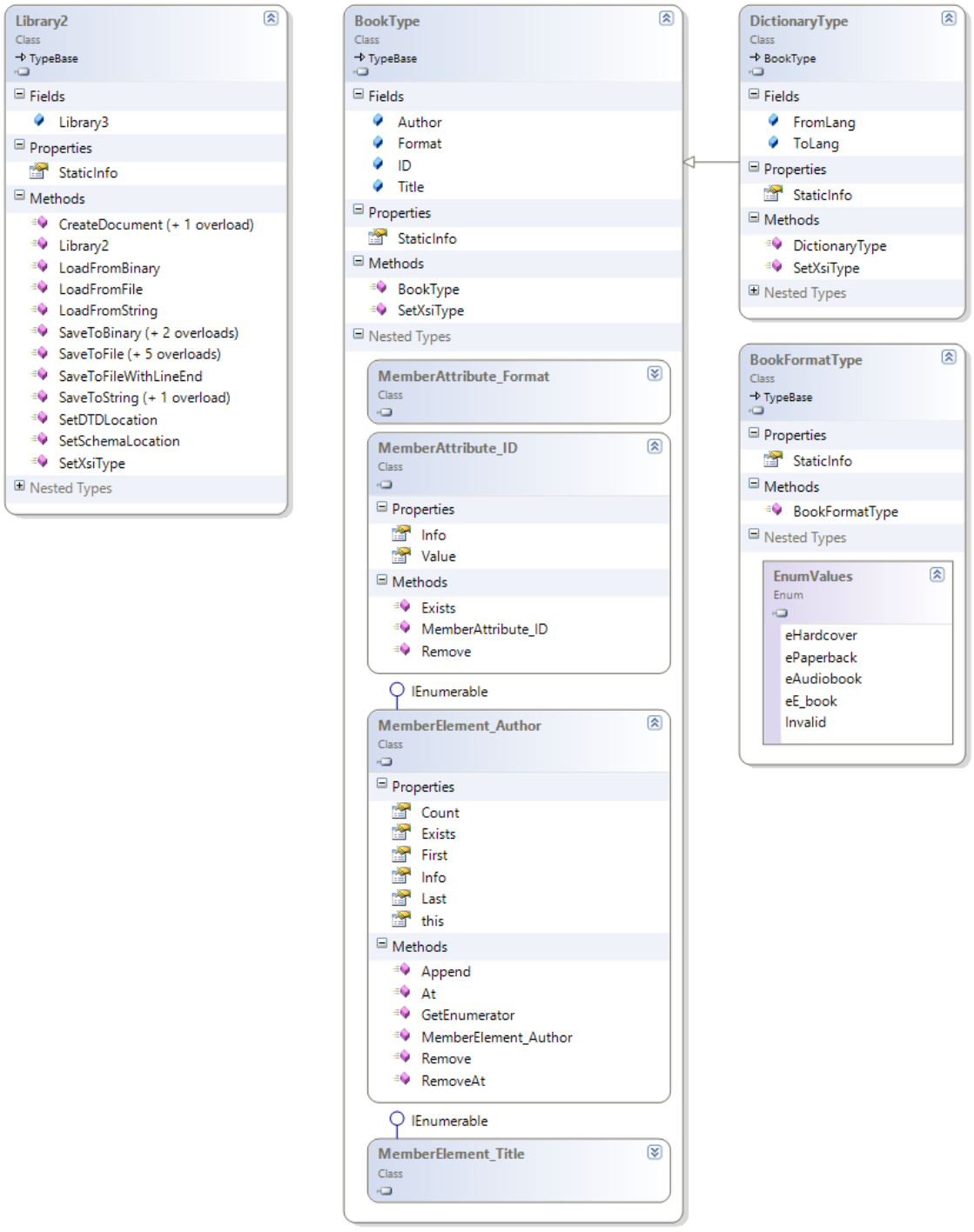
4. Press **F5** to start debugging.

16.7.5.3 Reading and Writing XML Documents (C#)

After you generate code from the Library schema (see Example Schema), a test C# application is created, along with several supporting Altova libraries.

About the generated C# libraries

The following diagram illustrates some of the most important classes of the generated code.



The central class of the generated code is the `Library2` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Library.xsd**, in this example). Note that this class is called `Library2` to avoid a possible conflict with the namespace name. As shown in the diagram, this class provides methods for loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a description of this class, see the class reference ([YourSchema].[Doc]).

The `Library3` member of the `Library2` class represents the actual root of the document. Again,

the number at the end is meant to avoid a naming conflict with the class name.

According to the code generation rules mentioned in About Schema Wrapper Libraries (C#), member classes are generated for each attribute and for each element of a type. In the generated code, the name of such member classes is prefixed with `MemberAttribute_` and `MemberElement_`, respectively. In the diagram above, examples of such classes are `MemberAttribute_ID` and `MemberElement_Author`, generated from the **Author** element and **ID** attribute of a book, respectively. Such classes enable you to manipulate programmatically the corresponding elements and attributes in the instance XML document (for example, append, remove, set value, etc). For more information, see `[YourSchemaType].MemberAttribute` and `[YourSchemaType].MemberElement` class reference.

Since the **DictionaryType** is a complex type derived from **BookType** in the schema, this relationship is also reflected in the generated classes. As illustrated in the diagram, the class `DictionaryType` inherits the `BookType` class.

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` that is a member of the `BookFormatType` class.

Writing an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio generated from the Library schema mentioned earlier in this example.

> While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

2. In Solution Explorer, open the **LibraryTest.cs** file, and edit the `Example()` method as shown below.

```
        protected static void Example()
        {
            // Create a new XML library
            Library2 doc = Library2.CreateDocument();
            // Append the root element
            LibraryType root = doc.Library3.Append();

            // Create the library generation date using Altova DateTime
class
            Altova.Types.DateTime dt = new
Altova.Types.DateTime(System.DateTime.Now);
            // Append the date to the root
            root.LastUpdated.Value = dt;

            // Add a new book
            BookType book = root.Book.Append();
            // Set the value of the ID attribute
            book.ID.Value = 1;
            // Set the format of the book (enumeration)
```

```
            book.Format.EnumerationValue =
BookFormatType.EnumValues.eHardcover;
            // Set the Title and Author elements
            book.Title.Append().Value = "The XMLSpy Handbook";
            book.Author.Append().Value = "Altova";

            // Append a dictionary (book of derived type) and populate its
attributes and elements
            DictionaryType dictionary = new
DictionaryType(root.Book.Append().Node);
            dictionary.ID.Value = 2;
            dictionary.Title.Append().Value = "English-German Dictionary";
            dictionary.Format.EnumerationValue =
BookFormatType.EnumValues.eE_book;
            dictionary.Author.Append().Value = "John Doe";
            dictionary.FromLang.Append().Value = "English";
            dictionary.ToLang.Append().Value = "German";
            // Since it's a derived type, make sure to set the xsi:type
attribute of the book element
            dictionary.SetXsiType();

            // Optionally, set the schema location (adjust the path if
            // your schema is not in the same folder as the generated
instance file)
            doc.SetSchemaLocation("Library.xsd");

            // Save the XML document with the "pretty print" option enabled
            doc.SaveToFile("GeneratedLibrary.xml", true);
        }
```

3. Press **F5** to start debugging. If the code was executed successfully, a **GeneratedLibrary.xml** file is created in the solution output directory (typically, **bin/Debug**).

Reading an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio.
2. Save the code below as **Library.xml** to the output directory of the project (by default, **bin/Debug**). This is the file that will be read by the program code.

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample" xsi:schemaLocation="http://
www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-02-
03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. In Solution Explorer, open the **LibraryTest.cs** file, and edit the `Example()` method as shown below.

```
       protected static void Example()
        {
            // Load the XML file into a new Library instance
            Library2 doc = Library2.LoadFromFile("Library.xml");
            // Get the root element
            LibraryType root = doc.Library3.First;

            // Read the library generation date
            Altova.Types.DateTime dt = root.LastUpdated.Value;
            string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
            Console.WriteLine("The library generation date is: " +
dt_as_string);

            // Iteration: for each <Book>...
            foreach (BookType book in root.Book)
            {
                // Output values of ID attribute and (first and only) title
element
                Console.WriteLine("ID:    " + book.ID.Value);
                Console.WriteLine("Title: " + book.Title.First.Value);

                // Read and compare an enumeration value
                if (book.Format.EnumerationValue ==
BookFormatType.EnumValues.ePaperback)
                    Console.WriteLine("This is a paperback book.");

                // Iteration: for each <Author>
                foreach (xs.stringType author in book.Author)
                    Console.WriteLine("Author: " + author.Value);

                // Determine if this book is of derived type
                if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
                {
                    // Find the value of the xsi:type attribute
                    string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
                    // Get the namespace URI and the lookup prefix of this
namespace
                    string namespaceUri = book.Node.NamespaceURI;
                    string prefix =
book.Node.GetPrefixOfNamespace(namespaceUri);

                    // if this book has DictionaryType
                    if (namespaceUri == "http://www.nanonull.com/
LibrarySample" && xsiTypeValue.Equals(prefix + ":DictionaryType"))
                    {
                        // output additional fields
                        DictionaryType dictionary = new
DictionaryType(book.Node);
                        Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
                        Console.WriteLine("Language to: " +
dictionary.ToLang.First.Value);
                    }
                    else
                    {
```

```
                        throw new Exception("Unexpected book type");
                    }
                }
            }

            Console.ReadLine();
        }
```

4. Press **F5** to start debugging. If the code was executed successfully, **Library.xml** will be read by the program code, and its contents displayed as console output.

Reading and writing elements and attributes

Values of attributes and elements can be accessed using the `Value` property of the generated member element or attribute class, for example:

```
// Output values of ID attribute and (first and only) title element
Console.WriteLine("ID:    " + book.ID.Value);
Console.WriteLine("Title:" + book.Title.First.Value);
```

To get the value of the **Title** element in this particular example, we also used the `First()` method, since this is the first (and only) **Title** element of a book. For cases when you need to pick a specific element from a list by index, use the `At()` method.

The class generated for each member element of a type implements the standard `System.Collections.IEnumerable` interface. This makes it possible to loop through multiple elements of the same type. In this particular example, you can loop through all books of a Library object as follows:

```
// Iteration: for each <Book> ...
foreach (BookType book in root.Book)
{
    // your code here ...
}
```

To add a new element, use the `Append()` method. For example, the following code appends the root element to the document:

```
// Append the root element to the library
LibraryType root = doc.Library3.Append();
```

You can set the value of an attribute (like ID in this example) as follows:

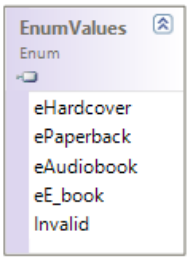
```
// Set the value of the ID attribute
book.ID.Value = 1;
```

For further information, see `[YourSchemaType].MemberAttribute` and `[YourSchemaType].MemberElement` class reference.

Reading and writing enumeration values

If your XML schema defines simple types as enumerations, the enumerated values become

available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum`:



To assign enumeration values to an object, use code such as the one below:

```
// Set the format of the book (enumeration)
book.FormatEnumerationValue = BookFormatType.EnumValues.eHardcover;
```

You can read such enumeration values from XML instance documents as follows:

```
// Read and compare an enumeration value
if (book.FormatEnumerationValue == BookFormatType.EnumValues.ePaperback)
Console.WriteLine("This is a paperback book.");
```

When an "if" condition is not enough, create a switch to determine each enumeration value and process it as required.

Working with xs:dateTime and xs:duration types

If the schema from which you generated code uses time and duration types such as `xs:dateTime`, or `xs:duration`, these are converted to Altova native classes in generated code. Therefore, to write a date or duration value to the XML document, do the following:

1. Construct an `Altova.Types.DateTime` or `Altova.Types.Duration` object (either from `System.DateTime`, or by using parts such as hours and minutes, see `Altova.Types.DateTime` and `Altova.Types.Duration` for more information).
2. Set the object as value of the required element or attribute, for example:

```
// Create the library generation date using Altova DateTime class
Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
// Append the date to the root
rootLastUpdatedValue = dt;
```

To read a date or duration from an XML document, do the following:

1. Declare the element value (or attribute) as `Altova.Types.DateTime` or `Altova.Types.Duration` object.
2. Format the required element or attribute, for example:

```
// Read the library generation date
Altova.Types.DateTime dt = rootLastUpdatedValue;
string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
Console.WriteLine("The library generation date is:" + dt_as_string);
```

For more information, see Altova.Types.DateTime and Altova.Types.Duration class reference.

Working with derived types

If your XML schema defines derived types, you can preserve type derivation in XML documents that you create or load programmatically. Taking the schema used in this example, the following code listing illustrates how to create a new book of derived type `DictionaryType`:

```
// Append a dictionary (book of derived type) and populate its attributes and elements
DictionaryType dictionary = new DictionaryType(rootBook.Append().Node);
dictionary.ID.Value = 2;
dictionary.Title.Append().Value = "English-German Dictionary";
dictionary.Author.Append().Value = "John Doe";
dictionary.FromLanguage.Append().Value = "English";
dictionary.ToLanguage.Append().Value = "German";

// Since it's a derived type, make sure to set the xsi:type attribute of the book element
dictionary.SetXsiType();
```

Note that it is important to set the `xsi:type` attribute of the newly created book. This ensures that the book type will be interpreted correctly by the schema when the XML document is validated.

When you load data from an XML document, the following code listing shows how to identify a book of derived type `DictionaryType` in the loaded XML instance. First, the code finds the value of the `xsi:type` attribute of the book node. If the namespace URI of this node is `http://www.nanonull.com/LibrarySample`, and if the URI lookup prefix and type matches the value of the `xsi:type` attribute, then this is a dictionary:

```
      // Determine if this book is of derived type
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
          // Find the value of the xsi:type attribute
          string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
          // Get the namespace URI and the lookup prefix of this namespace
          string namespaceUri = book.Node.NamespaceURI;
          string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

          // if this book has DictionaryType
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // output additional fields
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " +
dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

16.7.5.4 Reading and Writing XML Documents (Java)

After you generate code from the Library schema (see Example Schema), a test Java project is created, along with several supporting Altova libraries.

About the generated Java libraries

The following diagram illustrates some of the most important classes of the generated code.



The central class of the generated code is the `Library2` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Library.xsd**, in this example). Note that this class is called `Library2` to avoid a possible conflict with the namespace name. As shown in the diagram, this class provides methods for

loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a description of this class, see the class reference (`com.[YourSchema].[Doc]`).

The `Library3` member of the `Library2` class represents the actual root of the document. Again, the number at the end is meant to avoid a naming conflict with the class name.

According to the code generation rules mentioned in About Generated Java Code, member classes are generated for each attribute and for each element of a type. In the generated code, the name of such member classes is prefixed with `MemberAttribute_` and `MemberElement_`, respectively. In the diagram above, examples of such classes are `MemberAttribute_ID` and `MemberElement_Author`, generated from the **Author** element and **ID** attribute of a book, respectively. Such classes enable you to manipulate programmatically the corresponding elements and attributes in the instance XML document (for example, append, remove, set value, etc). For more information, see `com.[YourSchema].[YourSchemaType].MemberAttribute` and `com.[YourSchema].[YourSchemaType].MemberElement` class reference.

Since the **DictionaryType** is a complex type derived from **BookType** in the schema, this relationship is also reflected in the generated classes. As illustrated in the diagram, the class `DictionaryType` inherits the `BookType` class.

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` that is a member of the `BookFormatType` class.

Writing an XML document

1. On the **File** menu of Eclipse, click **Import**, select **Existing Projects into Workspace**, and click **Next**.
2. Next to **Select root directory**, click **Browse**, select the directory to which you generated the Java code, and then click **Finish**.
3. In the Eclipse Package Explorer, expand the **com.LibraryTest** package and open the **LibraryTest.java** file.

> While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

4. Edit the `Example()` method as shown below.

```
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();

      // set the "LastUpdated" attribute
      com.altova.types.DateTime dt = new
com.altova.types.DateTime(DateTime.now());
      lib.LastUpdated.setValue(dt);
```

```
      // create a new <Book> and populate its elements and attributes
      BookType book = lib.Book.append();
      book.ID.setValue(java.math.BigInteger.valueOf(1));
      book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
      book.Title.append().setValue("The XML Spy Handbook");
      book.Author.append().setValue("Altova");

      // create a dictionary (book of derived type)  and populate its
elements and attributes
      DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
      dict.ID.setValue(java.math.BigInteger.valueOf(2));
      dict.Title.append().setValue("English-German Dictionary");
      dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
      dict.Author.append().setValue("John Doe");
      dict.FromLang.append().setValue("English");
      dict.ToLang.append().setValue("German");
      dict.setXsiType();

      // set the schema location (this is optional)
      libDoc.setSchemaLocation("Library.xsd");

      // save the XML document to a file with default encoding (UTF-8).
"true" causes the file to be pretty-printed.
      libDoc.saveToFile("Library1.xml", true);
   }
```

5. Build the Java project and run it. If the code is executed successfully, a **Library1.xml** file is created in the project directory.

Reading an XML document

1. On the **File** menu of Eclipse, click **Import**, select **Existing Projects into Workspace**, and click **Next**.
2. Next to **Select root directory**, click **Browse**, select the directory to which you generated the Java code, and then click **Finish**.
3. Save the code below as **Library1.xml** to a local directory (you will need to refer to the path of the **Library1.xml** file from the sample code below).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample" xsi:schemaLocation="http://
www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-02-
03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

4. In the Eclipse Package Explorer, expand the **com.LibraryTest** package and open the **LibraryTest.java** file.
5. Edit the `Example()` method as shown below.

```
   protected static void example() throws Exception {
      // load XML document from a path, make sure to adjust the path as
necessary
      Library2 libDoc = Library2.loadFromFile("Library1.xml");

      // get the first (and only) root element <Library>
      LibraryType lib = libDoc.Library3.first();

      // check whether an element exists:
      if (!lib.Book.exists()) {
         System.out.println("This library is empty.");
         return;
      }

      // read a DateTime schema type
      com.altova.types.DateTime dt = lib.LastUpdated.getValue();
      System.out.println("The library was last updated on: " +
dt.toDateString());

      // iteration: for each <Book>...
      for (java.util.Iterator itBook = lib.Book.iterator();
itBook.hasNext();) {
         BookType book = (BookType) itBook.next();
         // output values of ID attribute and (first and only) title element
         System.out.println("ID: " + book.ID.getValue());
         System.out.println("Title: " + book.Title.first().getValue());

         // read and compare an enumeration value
         if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
            System.out.println("This is a paperback book.");

         // for each <Author>...
         for (java.util.Iterator itAuthor = book.Author.iterator(); itAuthor
               .hasNext();)
            System.out.println("Author: " + ((com.Library.xs.stringType)
itAuthor.next()).getValue());

         // find the derived type of this book
         // by looking at the value of the xsi:type attribute, using DOM

         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // Get the value of the xsi:type attribute
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // Get the namespace URI and lookup prefix of this namespace
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // If xsi:type matches the namespace URI and type of the book
node
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix +
":DictionaryType" )))    {
               // ...then this is a book of derived type (dictionary)
```

```
               DictionaryType dictionary = new
DictionaryType(   book.getNode());
               // output the value of the "FromLang" and "ToLang" elements
               System.out.println("From language: " +
dictionary.FromLang.first().getValue());
               System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // throw an error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
      }
   }
```

6. Build the Java project and run it. If the code is executed successfully, **Library1.xml** will be read by the program code, and its contents displayed in the Console view.

Reading and writing elements and attributes

Values of attributes and elements can be accessed using the `getValue()` method of the generated member element or attribute class, for example:

```
// output values of ID attribute and (first and only) title element
System.out.println("ID: " + book.ID.getValue());
System.out.println("Title: " + book.Title.first().getValue());
```

To get the value of the **Title** element in this particular example, we also used the `first()` method, since this is the first (and only) **Title** element of a book. For cases when you need to pick a specific element from a list by index, use the `at()` method.

To iterate through multiple elements, use either index-based iteration or `java.util.Iterator`. For example, you can iterate through the books of a library as follows:

```
// index-based iteration
for (int j = 0; j < lib.Book.count(); ++j ) {
   // your code here
}

// alternative iteration using java.util.Iterator
for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {

      // your code here
   }
```

To add a new element, use the `append()` method. For example, the following code appends an empty root **Library** element to the document:

```
// create the root element <Library> and add it to the document
LibraryType lib = libDoc.Library3.append();
```

Once an element is appended, you can set the value of any of its elements or an attributes by using the setValue() method.

```
// set the value of the Title element
book.Title.append().setValue("The XML Spy Handbook");
// set the value of the ID attribute
book.ID.setValue(java.math.BigInteger.valueOf(1));
```

For further information, see `com.[YourSchema].[YourSchemaType].MemberAttribute` and `com.[YourSchema].[YourSchemaType].MemberElement` class reference.

Reading and writing enumeration values

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` (see the `BookFormatType` class diagram above). To assign enumeration values to an object, use code such as the one below:

```
// set an enumeration value
book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
```

You can read such enumeration values from XML instance documents as follows:

```
// read an enumeration value
if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
        System.out.println("This is a paperback book."
```

When an "if" condition is not enough, create a switch to determine each enumeration value and process it as required.

Working with xs:dateTime and xs:duration types

If the schema from which you generated code uses time and duration types such as `xs:dateTime`, or `xs:duration`, these are converted to Altova native classes in generated code. Therefore, to write a date or duration value to the XML document, do the following:

1. Construct a `com.altova.types.DateTime` or `com.altova.types.Duration` object.
2. Set the object as value of the required element or attribute, for example:

```
// set the value of an attribute of DateTime type
com.altova.types.DateTime dt = new
com.altova.types.DateTime(DateTime.now());
lib.LastUpdated.setValue(dt);
```

To read a date or duration from an XML document:

1. Declare the element value (or attribute) as `com.altova.types.DateTime` or `com.altova.types.Duration` object.
2. Format the required element or attribute, for example:

```
// read a DateTime type
```

```
com.altova.types.DateTime dt = lib.LastUpdated.getValue();
   System.out.println("The library was last updated on: " +
dt.toDateString());
```

For more information, see com.altova.types.DateTime and com.altova.types.Duration class reference.

Working with derived types

If your XML schema defines derived types, you can preserve type derivation in XML documents that you create or load programmatically. Taking the schema used in this example, the following code listing illustrates how to create a new book of derived type `DictionaryType`:

```
// create a dictionary (book of derived type)  and populate its elements and
attributes
DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
dict.ID.setValue(java.math.BigInteger.valueOf(2));
dict.Title.append().setValue("English-German Dictionary");
dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
dict.Author.append().setValue("John Doe");
dict.FromLang.append().setValue("English");
dict.ToLang.append().setValue("German");
dict.setXsiType();
```

Note that it is important to set the `xsi:type` attribute of the newly created book. This ensures that the book type will be interpreted correctly by the schema when the XML document is validated.

When you load data from an XML document, the following code listing shows how to identify a book of derived type `DictionaryType` in the loaded XML instance. First, the code finds the value of the `xsi:type` attribute of the book node. If the namespace URI of this node is `http://www.nanonull.com/LibrarySample`, and if the URI lookup prefix and type matches the value of the `xsi:type` attribute, then this is a dictionary:

```
         // find the derived type of this book
         // by looking at the value of the xsi:type attribute, using DOM

         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // Get the value of the xsi:type attribute
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // Get the namespace URI and lookup prefix of the book node
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // If xsi:type matches the namespace URI and type of the book
node
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix +
":DictionaryType" )))    {
               // ...then this is a book of derived type (dictionary)

               DictionaryType dictionary = new
DictionaryType(   book.getNode());
               // output the value of the "FromLang" and "ToLang" elements
```

```
                System.out.println("From language: " +
dictionary.FromLang.first().getValue());
                System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // throw an error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
```

16.8 Reference to Generated Classes (C++)

This chapter includes a description of C++ classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code may include other supporting classes, which are not listed here and are subject to modification.

16.8.1 altova::DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

Name	Description
`DateTime()`	Initializes a new instance of the `DateTime` class to 12:00:00 midnight, January 1, 0001.
`DateTime(__int64 value, short timezone)`	Initializes a new instance of the `DateTime` class. The `value` parameter represents the number of ticks (100-nanosecond intervals) that have elapsed since 12:00:00 midnight, January 1, 0001.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, and second supplied as argument.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second, short timezone)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second and timezone supplied as argument. The timezone is expressed in minutes and can be positive or negative. For example, the timezone "UTC-01:00" is expressed as "-60".

Methods

Name	Description
`unsigned char Day() const`	Returns the day of month of the current `DateTime` object. The return values range from 1 through 31.
`int DayOfYear() const`	Returns the day of year of the current `DateTime` object. The return values range from 1 through 366.
`bool HasTimezone() const`	Returns Boolean **true** if the current `DateTime` object has a timezone defined; **false** otherwise.
`unsigned char Hour() const`	Returns the hour of the current `DateTime` object. The

Name	Description
	return values range from 0 through 23.
`static bool IsLeapYear(int year)`	Returns Boolean **true** if the year of the `DateTime` class is a leap year; **false** otherwise.
`unsigned char Minute() const`	Returns the minute of the current `DateTime` object. The return values range from 0 through 59.
`unsigned char Month() const`	Returns the month of the current `DateTime` object. The return values range from 1 through 12.
`__int64 NormalizedValue() const`	Returns the value of the `DateTime` object expressed as the Coordinated Universal Time (UTC).
`double Second() const`	Returns the second of the current `DateTime` object. The return values range from 0 through 59.
`void SetTimezone(short tz)`	Sets the timezone of the current `DateTime` object to the timezone value supplied as argument. The **tz** argument is expressed in minutes and can be positive or negative.
`short Timezone() const`	Returns the timezone, in minutes, of the current `DateTime` object. Before using this method, make sure that the object actually has a timezone, by calling the `HasTimezone()` method.
`__int64 Value() const`	Returns the value of the `DateTime` object, expressed in the number of ticks (100-nanosecond intervals) that have elapsed since 12:00:00 midnight, January 1, 0001.
`int Weekday() const`	Returns the day of week of the current `DateTime` object, as an integer. Values range from 0 through 6, where 0 is Monday (ISO-8601).
`int Weeknumber() const`	Returns the number of week in the year of the current `DateTime` object. The return values are according to ISO-8601.
`int WeekOfMonth() const`	Returns the number of week in the month of the current `DateTime` object. The return values are according to ISO-8601.
`int Year() const`	Returns the year of the current `DateTime` object.

Example

```
void Example()
{
	// initialize a new DateTime instance to 12:00:00 midnight, January 1st, 0001
	altova:DateTime dt1 = altova:DateTime();

	// initialize a new DateTime instance using the year, month, day, hour, minute, and
second
	altova:DateTime dt2 = altova:DateTime(2015,11,10,9,8,7);

	// initialize a new DateTime instance using the year, month, day, hour, minute, second,
```

```
and UTC +01:00 timezone
        altova:DateTime dt = altova:DateTime(2015,11,22,13,53,7,60);

       // Get the value of this DateTime object
       std::cout << "The number of ticks of the DateTime object is:" << dtValue() <<
std::endl;

       // Get the year
       cout << "The year is:" << dtYear() << endl;
       // Get the month
       cout << "The month is:" << (int)dtMonth() << endl;
       // Get the day of the month
       cout << "The day of the month is:" << (int) dtDay() << endl;
       // Get the day of the year
       cout << "The day of the year is:" << dtDayOfYear() << endl;
       // Get the hour
       cout << "The hour is:" << (int) dtHour() << endl;
       // Get the minute
       cout << "The minute is:" << (int) dtMinute() << endl;
       // Get the second
       cout << "The second is:" << dtSecond() << endl;
       // Get the weekday
       cout << "The weekday is:" << dtWeekday() << endl;
       // Get the week number
       cout << "The week of year is:" << dtWeeknumber() << endl;
       // Get the week in month
       cout << "The week of month is:" << dtWeekOfMonth() << endl;

       // Check whether a DateTime instance has a timezone
       if (dtHasTimezone() == TRUE)
       {
              // output the value of the Timezone
              cout <<  "The timezone is:" << dtTimezone() << endl;
       }
       else
       {
              cout <<  "No timezone has been defined." << endl;
       }

       // Construct a DateTime object with a timezone UTC+01:00 (Vienna)
       altova:DateTime vienna_dt = DateTime(2015,11,23,14,30,59,+60);
       // Output the result in readable format
       cout << "The Vienna time:"
              << (int) vienna_dtMonth()
              << "-" << (int) vienna_dtDay()
              << " " << (int) vienna_dtHour()
              << ":" << (int) vienna_dtMinute()
              << ":" << (int) vienna_dtSecond()
              << endl;

       // Convert the value to UTC time
       DateTime utc_dt = DateTime(vienna_dtNormalizedValue());
       // Output the result in readable format
       cout << "The UTC time:  "
              << (int) utc_dtMonth()
              << "-" << (int) utc_dtDay()
```

```
                << " " << (int)utc_dtHour()
                << ":" << (int)utc_dtMinute()
                << ":" << (int)utc_dtSecond()
                << endl;

        // Check if a year is a leap year
        int year = 2016;
        if(altova:DateTime::IsLeapYear(year))
        {cout << year << " is a leap year" << endl;}
        else
        {cout << year << " is not a leap year" << endl;}
}
```

16.8.2 altova::Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

Name	Description
`Duration()`	Initializes a new instance of the `Duration` class to an empty value.
`Duration(const DayTimeDuration& dt)`	Initializes a new instance of the `Duration` class to a duration defined by the `dt` argument (see altova::DayTimeDuration).
`Duration(const YearMonthDuration& ym )`	Initializes a new instance of the `Duration` class to the duration defined by the `ym` argument (see altova::YearMonthDuration).
`Duration(const YearMonthDuration& ym, const DayTimeDuration& dt)`	Initializes a new instance of the `Duration` class to the duration defined by both the `dt` and the `ym` arguments (see altova::YearMonthDuration and altova::DayTimeDuration).

Methods

Name	Description
`int Days() const`	Returns the number of days in the current `Duration` instance.
`DayTimeDuration DayTime() const`	Returns the day and time duration in the current `Duration` instance, expressed as a `DayTimeDuration` object (see altova::DayTimeDuration).
`int Hours() const`	Returns the number of hours in the current `Duration` instance.
`bool IsNegative() const`	Returns Boolean **true** if the current `Duration` instance is negative.

Name	Description
`bool IsPositive() const`	Returns Boolean **true** if the current `Duration` instance is positive.
`int Minutes() const`	Returns the number of minutes in the current `Duration` instance.
`int Months() const`	Returns the number of months in the current `Duration` instance.
`double Seconds() const`	Returns the number of seconds in the current `Duration` instance.
`YearMonthDuration YearMonth() const`	Returns the year and month duration in the current `Duration` instance, expressed as a `YearMonthDuration` object (see altova::YearMonthDuration).
`int Years() const`	Returns the number of years in the current `Duration` instance.

Example

The following code listing illustrates creating a new `Duration` object, as well as reading values from it.

```
void ExampleDuration()
{
	// Create an empty Duration object
	altova:Duration empty_duration = altova:Duration();

	// Create a Duration object using an existing duration value
	altova:Duration duration1 = altova:Duration(empty_duration);

	// Create a YearMonth duration of six years and five months
	altova::YearMonthDuration yrduration = altova::YearMonthDuration(6,5);

	// Create a DayTime duration of four days, three hours, two minutes, and one second
	altova:DayTimeDuration dtduration = altova:DayTimeDuration(4,3,2,1);

	// Create a Duration object by combining the two previously created durations
	altova:Duration duration = altova:Duration(yrduration,dtduration);

	// Get the number of years in this Duration instance
	cout << "Years: " << duration.Years() << endl;

	// Get the number of months in this Duration instance
	cout << "Months:" << duration.Months() << endl;

	// Get the number of days in this Duration instance
	cout << "Days: " << duration.Days() << endl;

	// Get the number of hours in this Duration instance
	cout << "Hours: " << duration.Hours() << endl;

	// Get the number of hours in this Duration instance
```

```
        cout << "Minutes:" << durationMinutes() << endl;

        // Get the number of seconds in this Duration instance
        cout << "Seconds:" << durationSeconds() << endl;
}
```

16.8.3 altova::DayTimeDuration

This class enables you to process XML schema duration types that consist of a day and time part.

Constructors

Name	Description
`DayTimeDuration()`	Initializes a new instance of the `DayTimeDuration` class to an empty value.
`DayTimeDuration(int days, int hours, int minutes, double seconds)`	Initializes a new instance of the `DayTimeDuration` class to the number of days, hours, minutes, and seconds supplied as arguments.
`explicit DayTimeDuration(__int64 value)`	Initializes a new instance of the `DayTimeDuration` class to a duration that consists of as many ticks (100-nanosecond intervals) as supplied in the **value** argument.

Methods

Name	Description
`int Days() const`	Returns the number of days in the current `DayTimeDuration` instance.
`int Hours() const`	Returns the number of hours in the current `DayTimeDuration` instance.
`bool IsNegative() const`	Returns Boolean **true** if the current `DayTimeDuration` instance is negative.
`bool IsPositive() const`	Returns Boolean **true** if the current `DayTimeDuration` instance is positive.
`int Minutes() const`	Returns the number of minutes in the current `DayTimeDuration` instance.
`double Seconds() const`	Returns the number of seconds in the current `DayTimeDuration` instance.
`__int64 Value() const`	Returns the value (in ticks) of the current `DayTimeDuration` instance.

16.8.4 altova::YearMonthDuration

This class enables you to process XML schema duration types that consist of a year and month part.

Constructors

Name	Description
`YearMonthDuration()`	Initializes a new instance of the `YearMonthDuration` class to an empty value.
`YearMonthDuration(int years, int months)`	Initializes a new instance of the `YearMonthDuration` class to the number of years and months supplied in the **years** and **months** arguments.
`explicit YearMonthDuration(int value)`	Initializes a new instance of the `YearMonthDuration` class to a duration that consists of as many ticks (100-nanosecond intervals) as supplied in the **value** argument.

Methods

Name	Description
`bool IsNegative() const`	Returns Boolean **true** if the current `YearMonthDuration` instance is negative.
`bool IsPositive() const`	Returns Boolean **true** if the current `YearMonthDuration` instance is positive.
`int Months() const`	Returns the number of months in the current `YearMonthDuration` instance.
`int Value() const`	Returns the value (in ticks) of the current `YearMonthDuration` instance.
`int Years()`	Returns the number of years in the current `YearMonthDuration` instance.

16.8.5 altova::meta::Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

Methods

Name	Description
`SimpleType GetDataType()`	Returns the type of the attribute content.
`string_type GetLocalName()`	Returns the local name of the attribute.
`string_type GetNamespaceURI()`	Returns the namespace URI of the attribute.
`bool IsRequired()`	Returns true if the attribute is required.

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL Attribute.
`bool operator!()`	Returns true if this is the NULL Attribute.

16.8.6 altova::meta::ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Methods

Name	Description
`Attribute FindAttribute(const char_type* localName, const char_type* namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
`Element FindElement(const char_type* localName, const char_type* namespaceURI)`	Finds the element with the specified local name and namespace URI.
`std::vector<Attribute> GetAttributes()`	Returns a list of all attributes.
`ComplexType GetBaseType()`	Returns the base type of this type.
`SimpleType GetContentType()`	Returns the simple type of the content.
`std::vector<Element> GetElements()`	Returns a list of all elements.

Name	Description
`string_type GetLocalName()`	Returns the local name of the type.
`string_type GetNamespaceURI()`	Returns the namespace URI of the type.

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL ComplexType.
`bool operator!()`	Returns true if this is the NULL ComplexType.

16.8.7 altova::meta::Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

Methods

Name	Description
`ComplexType GetDataType()`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use GetContentType() of the returned object to get the simple content type.
`string_type GetLocalName()`	Returns the local name of the element.
`unsigned int GetMaxOccurs()`	Returns the maxOccurs value defined in the schema.
`unsigned int GetMinOccurs()`	Returns the minOccurs value defined in the schema.
`string_type GetNamespaceURI()`	Returns the namespace URI of the element.

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL Element.
`bool operator!()`	Returns true if this is the NULL Element.

16.8.8 altova::meta::SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Methods

Name	Description
`SimpleType GetBaseType()`	Returns the base type of this type.
`std::vector<string_type> GetEnumerations()`	Returns a list of all enumeration facets.
`unsigned int GetFractionDigits()`	Returns the value of this facet.
`unsigned int GetLength()`	Returns the value of this facet.
`string_type GetLocalName()`	Returns the local name of the type.
`string_type GetMaxExclusive()`	Returns the value of this facet.
`string_type GetMaxInclusive()`	Returns the value of this facet.
`unsigned int GetMaxLength()`	Returns the value of this facet.
`string_type GetMinExclusive()`	Returns the value of this facet.
`string_type GetMinInclusive()`	Returns the value of this facet.
`unsigned int GetMinLength()`	Returns the value of this facet.
`string_type GetNamespaceURI()`	Returns the namespace URI of the type.
`std::vector<string_type> GetPatterns()`	Returns a list of all pattern facets.
`unsigned int GetTotalDigits()`	Returns the value of this facet.
`WhitespaceType GetWhitespace()`	Returns the value of the whitespace facet, which is one of: • Whitespace_Unknown • Whitespace_Preserve • Whitespace_Replace • Whitespace_Collapse

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL SimpleType.
`bool operator!()`	Returns true if this is the NULL SimpleType.

16.8.9 [YourSchema]::[CDoc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the following methods. Note that, in the method names below, "CDoc" stands for the name of the generated document class itself.

Methods

Name	Description
`static CDoc CreateDocument()`	Creates a new, empty XML document. Must be released using DestroyDocument().
`void DestroyDocument()`	Destroys a document. All references to the document and its nodes are invalidated. This must be called when you finished working with a document.
`static CDoc LoadFromBinary(const std:vector<unsigned char>& xml)`	Loads an XML document from a byte array.
`static CDoc LoadFromFile(const string_type& fileName)`	Loads an XML document from a file.
`static CDoc LoadFromString(const string_type& xml)`	Loads an XML document from a string.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint)`	Saves an XML document to a byte array. When set to true, the `prettyPrint` argument re-formats the XML document for better readability.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
`void SaveToFile(const string_type & fileName,`	Saves an XML document to a file, with optional

Name	Description
`bool prettyPrint)`	"pretty-print" formatting.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl)`	Saves an XML document to a file. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl, const string_type & encoding)`	Saves an XML document to a file with the specified encoding. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. Byte order and Unicode byte-order mark can be specified for Unicode encodings. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`void SaveToFile(const string_type& fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`void SaveToFile(const string_type & fileName, bool prettyPrint, const string_type & encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Saves an XML document to a file with the specified encoding and the specified line end. Byte order and Unicode byte-order mark can be specified for Unicode encodings. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).

Name	Description
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`string_type SaveToString(bool prettyPrint)`	Saves an XML document to a string, with optional "pretty-print" formatting.
`string_type SaveToString(bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a string, with optional "pretty-print" formatting. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SetDTDLocation(const string_type & dtdLocation)`	Adds a DOCTYPE declaration with the specified system ID. A root element must already exist. This method is not supported for MSXML, since it is not possible to add a DOCTYPE declaration to a document in memory.
`void SetSchemaLocation(const string_type & schemaLocation)`	Adds an xsi:schemaLocation or xsi:noNamespaceSchemaLocation attribute to the root element. A root element must already exist.

16.8.10 [YourSchema]::MemberAttribute

When code is generated from an XML schema, a class such as this one is created for each member attribute of a type.

Methods

Name	Description
`bool exists()`	Returns true if the attribute exists.
`int GetEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or "Invalid" if the value does not match any of the enumerated values in the schema.
`altova::meta::Attribute info()`	Returns an object for querying schema information (see `altova::meta::Attribute`).
`void remove()`	Removes the attribute from its parent element.
`void SetEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.

16.8.11 [YourSchema]::MemberElement

When code is generated from an XML schema, a class such as this one is created for each member element of a type. In the descriptions below, "MemberType" stands for the name of the member element itself.

Methods

Name	Description
`Iterator<MemberType> all()`	Returns an object for iterating instances of the member element.
`MemberType append()`	Creates a new element and appends it to its parent.
`unsigned int count()`	Returns the count of elements.
`int GetEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
`bool exists()`	Returns true if at least one element exists.
`MemberType first()`	Returns the first instance of the member element.
`MemberType operator[] (unsigned int index)`	Returns the member element specified by the index.
`altova::meta::Element info()`	Returns an object for querying schema information (see `altova::meta::Element`).
`MemberType last()`	Returns the last instance of the member element.
`void remove()`	Deletes all occurrences of the element from its parent.
`void remove(unsigned int index)`	Deletes the occurrence of the element specified by the index.
`void SetEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.

16.9 Reference to Generated Classes (C#)

This chapter includes a description of C# classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code may include other supporting classes, which are not listed here and are subject to modification.

16.9.1 Altova.Types.DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

	Name	Description
	`DateTime(DateTime obj)`	Initializes a new instance of the `DateTime` class to the `DateTime` object supplied as argument.
	`DateTime(System.DateTime newvalue)`	Initializes a new instance of the `DateTime` class to the `System.DateTime` object supplied as argument.
	`DateTime(int year, int month, int day, int hour, int minute, double second, int offsetTZ)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, and timezone offset supplied as arguments.
	`DateTime(int year, int month, int day, int hour, int minute, double second)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, and second supplied as arguments.
	`DateTime(int year, int month, int day)`	Initializes a new instance of the `DateTime` class to the year, month and day supplied as arguments.

Properties

	Name	Description
	`bool HasTimezone`	Gets a Boolean value which indicates if the `DateTime` has a timezone.
	`static DateTime Now`	Gets a `DateTime` object that is set to the current date and time on this computer.
	`short TimezoneOffset`	Gets or sets the timezone offset, in minutes, of the `DateTime` object.
	`System.DateTime Value`	Gets or sets the value of the `DateTime` object as a `System.DateTime` value.

Methods

	Name	Description
	`int CompareTo(object obj)`	The `DateTime` class implements the `IComparable` interface. This method compares the current instance of `DateTime` to another object and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object. See also https://msdn.microsoft.com/en-us/library/system.icomparable.compareto(v=vs.110).aspx
	`override bool Equals(object obj)`	Returns **true** if the specified object is equal to the current object; **false** otherwise.
	`System.DateTime GetDateTime(bool correctTZ)`	Returns a `System.DateTime` object from the current `Altova.Types.DateTime` instance. The `correctTZ` Boolean argument specifies whether the time of the returned object must be adjusted according to the timezone of the current `Altova.Types.DateTime` instance.
	`override int GetHashCode()`	Returns the hash code of the current instance.
	`int GetWeekOfMonth()`	Returns the number of the week in month as an integer.

	`static DateTime Parse(string s)`	Creates a `DateTime` object from the string supplied as argument. For example, the following sample string values would be converted successfully to a `DateTime` object:
		
		`2015-01-01T23:23:23`
		`2015-01-01`
		`2015-11`
		`23:23:23`
		
		An exception is raised if the string cannot be converted to a `DateTime` object.
		
		Note that this method is static and can only be called on the `Altova.Types.DateTime` class itself, not on an instance of the class.

	`static DateTime Parse(string s,DateTimeFormat format)`	Creates a `DateTime` object from a string, using the format supplied as argument. For the list of possible formats, see Altova.Types.DateTimeFormat. An exception is raised if the string cannot be converted to a `DateTime` object. Note that this method is static and can only be called on the `Altova.Types.DateTime` class itself, not on an instance of the class.
	`override string ToString()`	Converts the `DateTime` object to a string.
	`string`	Converts the `DateTime` object to a string, using the

	Name	Description
	ToString(DateTimeFormat format)	format supplied as argument. For the list of possible formats, see Altova.Types.DateTimeFormat.

Operators

Name	Description
!=	Determines if DateTime a is not equal to DateTime b.
<	Determines if DateTime a is less than DateTime b.
<=	Determines if DateTime a is less than or equal to DateTime b.
==	Determines if DateTime a is equal to DateTime b.
>	Determines if DateTime a is greater than DateTime b.
>=	Determines if DateTime a is greater than or equal to DateTime b.

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
using Altova.Types;
```

The following code listing illustrates various ways to create DateTime objects:

```
protected static void DateTimeExample1()
{
        // Create a DateTime object from the current system time
        Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
        Console.WriteLine("The current time is:" + dt.ToString());

        // Create an Altova DateTime object from parts (no timezone)
        Altova.Types.DateTime dt1 = new Altova.Types.DateTime(2015,10,12,10,50,33);
        Console.WriteLine("My custom time is :" + dt1.ToString());

        // Create an Altova DateTime object from parts (with UTC+60 minutes timezone)
        Altova.Types.DateTime dt2 = new Altova.Types.DateTime(2015,10,12,10,50,33,60);
        Console.WriteLine("My custom time with timezone is :" + dt2.ToString());

        // Create an Altova DateTime object by parsing a string
        Altova.Types.DateTime dt3 = Altova.Types.DateTime.Parse("2015-01-01T23:23:23");
        Console.WriteLine("Time created from string:" + dt3.ToString());

        // Create an Altova DateTime object by parsing a string formatted as schema date
        Altova.Types.DateTime dt4 = Altova.Types.DateTime.Parse("2015-01-01",
DateTimeFormatW3_date);
        Console.WriteLine("Time created from string formatted as schema date:" +
dt4.ToString());
```

```
}
```

The following code listing illustrates various ways to format `DateTime` objects:

```
protected static void DateTimeExample2()
{
        // Create a DateTime object from the current system time
        Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);

        // Output the unformatted DateTime
        Console.WriteLine("Unformatted time:" + dt.ToString());

        // Output this DateTime formatted using various formats
        Console.WriteLine("S_DateTime:     " + dt.ToString(DateTimeFormat.S_DateTime));
        Console.WriteLine("S_Days:         " + dt.ToString(DateTimeFormat.S_Days));
        Console.WriteLine("S_Seconds:      " + dt.ToString(DateTimeFormat.S_Seconds));
        Console.WriteLine("W3_date:        " + dt.ToString(DateTimeFormat.W3_date));
        Console.WriteLine("W3_dateTime:    " + dt.ToString(DateTimeFormat.W3_dateTime));
        Console.WriteLine("W3_gDay:        " + dt.ToString(DateTimeFormat.W3_gDay));
        Console.WriteLine("W3_gMonth:      " + dt.ToString(DateTimeFormat.W3_gMonth));
        Console.WriteLine("W3_gMonthDay:   " +
dt.ToString(DateTimeFormat.W3_gMonthDay));
        Console.WriteLine("W3_gYear:       " + dt.ToString(DateTimeFormat.W3_gYear));
        Console.WriteLine("W3_gYearMonth:  " +
dt.ToString(DateTimeFormat.W3_gYearMonth));
        Console.WriteLine("W3_time:        " + dt.ToString(DateTimeFormat.W3_time));
}
```

16.9.2 Altova.Types.DateTimeFormat

The `DateTimeFormat` enum type has the following constant values:

Value	Description	Example
S_DateTime	Formats the value as standard dateTime, with a precision of a ten-millionth of a second, including timezone.	`2015-11-12 12:19:03.9019132 +01:00`
S_Days	Formats the value as number of days elapsed since the UNIX epoch.	`735913.631897345108796296 2963`
S_Seconds	Formats the value as number of seconds elapsed since the UNIX epoch, with a precision of a ten-millionth of a second.	`63582937678.0769062`
W3_date	Formats the value as schema date.	`2015-11-12`
W3_dateTime	Formats the value as schema dateTime.	`2015-11-12T15:12:14.5194251`

Value	Description	Example
W3_gDay	Formats the value as schema gDay.	`---12` (assuming that the date is 12th of the month)
W3_gMonth	Formats the value as schema gMonth.	`--11` (assuming that the month is November)
W3_gMonthDay	Formats the value as schema gMonthDay.	`--11-12` (assuming that the date is 12th of November)
W3_gYear	Formats the value as schema gYear.	`2015` (assuming that the year is 2015)
W3_gYearMonth	Formats the value as schema gYearMonth.	`2015-11` (assuming that the year is 2015 and the month is November)
W3_time	Formats the value as schema time, with a precision of a ten-millionth of a second.	`15:19:07.5582719`

16.9.3 Altova.Types.Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

	Name	Description
	`Duration(Duration obj)`	Initializes a new instance of the `Duration` class to the `Duration` object supplied as argument.
	`Duration(System.TimeSpan newvalue)`	Initializes a new instance of the `Duration` class to the `System.TimeSpan` object supplied as argument.
	`Duration(long ticks)`	Initializes a new instance of the `Duration` class to the number of ticks supplied as argument.
	`Duration(int newyears, int newmonths, int days, int hours, int minutes, int seconds, double partseconds, bool bnegative)`	Initializes a new instance of the `Duration` class to a duration built from parts supplied as arguments.

Properties

	Name	Description
	`int Months`	Gets or sets the number of months of the current instance of `Duration`.

	Name	Description
	`System.TimeSpan Value`	Gets or sets the value (as `System.TimeSpan`) of the current instance of `Duration`.
	`int Years`	Gets or sets the number of years of the current instance of `Duration`.

Methods

	Name	Description
	`override bool Equals(object other)`	Returns **true** if the specified object is equal to the current object; **false** otherwise.
	`override int GetHashCode()`	Returns the hash code of the current instance.
	`bool IsNegative()`	Returns **true** if the current instance of `Duration` represents a negative duration.
	`static Duration Parse(string s, ParseType pt)`	Returns an `Altova.Types.Duration` object parsed from the string supplied as argument, using the parse type supplied as argument. Valid parse type values: **DURATION** Parse duration assuming that year, month, day, as well as time duration parts exist. **YEARMONTH** Parse duration assuming that only year and month parts exist. **DAYTIME** Parse duration assuming that only the day and time parts exist. Note that this method is static and can only be called on the class itself, not on an instance of the class.
	`override string ToString()`	Converts the current `Duration` instance to string. For example, a time span of 3 hours, 4 minutes, and 5 seconds would be converted to "`PT3H4M5S`".
	`string ToYearMonthString()`	Converts the current `Duration` instance to string, using the "Year and Month" parse type.

Operators

Name	Description
`!=`	Determines if `Duration` a is not equal to `Duration` b.
`==`	Determines if `Duration` a is equal to `Duration` b.

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
using Altova.Types;
```

The following code listing illustrates various ways to create `Duration` objects:

```
protected static void DurationExample1()
{
        // Create a new time span of 3 hours, 4 minutes, and 5 seconds
        System.TimeSpan ts = new TimeSpan(3,4,5);
        // Create a Duration from the time span
        Duration dr = new Duration(ts);
        // The output is: PT3H4M5S
        Console.WriteLine("Duration created from TimeSpan:" + dr.ToString());

        // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3 hours,
    // 2 minutes, 1 second, and 33 of a second
        Duration dr1 = new Duration(6,5,4,3,2,1,33,true);
        // The output is: -P6Y5M4DT3H2M1.33S
        Console.WriteLine("Duration created from parts:" + dr1.ToString());

        // Create a Duration from a string using the DAYTIME parse type
        Duration dr2 = Altova.Types.Duration.Parse("-P4DT3H2M1S",
Duration.ParseType.DAYTIME);
        // The output is -P4DT3H2M1S
        Console.WriteLine("Duration created from string:" + dr2.ToString());

        // Create a duration from ticks
        Duration dr3 = new Duration(System.DateTime.UtcNow.Ticks);
        // Output the result
        Console.WriteLine("Duration created from ticks:" + dr3.ToString());
}
```

The following code listing illustrates getting values from `Duration` objects:

```
protected static void DurationExample2()
{
        // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3 hours,
    // 2 minutes, 1 second, and 33 of a second
        Duration dr = new Duration(6,5,4,3,2,1,33,true);
        // The output is: -P6Y5M4DT3H2M1.33S
        Console.WriteLine("The complete duration is:" + dr.ToString());

        // Get only the year and month part as string
        string dr1 = dr.ToYearMonthString();
        Console.WriteLine("The YEARMONTH part is:" + dr1);

        // Get the number of years in duration
        Console.WriteLine("Years:" + dr.Years);

        // Get the number of months in duration
        Console.WriteLine("Months:" + dr.Months);
}
```

16.9.4 Altova.Xml.Meta.Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

Properties

	Name	Description
	`SimpleType DataType`	Returns the type of the attribute content.
	`string LocalName`	Returns the local name of the attribute.
	`string NamespaceURI`	Returns the namespace URI of the attribute.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of the attribute.
	`bool Required()`	Returns true if the attribute is required.

16.9.5 Altova.Xml.Meta.ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Properties

	Name	Description
	`Attribute[] Attributes`	Returns a list of all attributes.
	`ComplexType BaseType`	Returns the base type of this type or null if no base type exists.
	`SimpleType ContentType`	Returns the simple type of the content.
	`Element[] Elements`	Returns a list of all elements.
	`string LocalName`	Returns the local name of the type.
	`string NamespaceURI`	Returns the namespace URI of the type.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of this type.

Methods

	Name	Description
	`ComplexType BaseType`	Returns the base type of this type.
	`bool Equals(obj)`	Checks if two info objects refer to the same type, based on qualified name comparison. Returns true if the type has the same qualified name.
	`Attribute FindAttribute(string localName, string namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
	`Element FindElement(string localName, string namespaceURI)`	Finds the element with the specified local name and namespace URI.

16.9.6 Altova.Xml.Meta.Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

Properties

	Name	Description
	`ComplexType DataType`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use the `ContentType` property of the returned object to get the simple content type.
	`string LocalName`	Returns the local name of the element.
	`int MaxOccurs`	Returns the `maxOccurs` value defined in the schema.
	`int MinOccurs`	Returns the `minOccurs` value defined in the schema.
	`string NamespaceURI`	Returns the namespace URI of the element.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of the element.

16.9.7 Altova.Xml.Meta.SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Properties

	Name	Description
	`SimpleType BaseType`	Returns the base type of this type.
	`string[] Enumerations`	Returns a list of all enumeration facets.
	`int FractionDigits`	Returns the value of this facet.
	`int Length`	Returns the value of this facet.
	`string LocalName`	Returns the local name of the type.
	`string MaxExclusive`	Returns the value of this facet.
	`string MaxInclusive`	Returns the value of this facet.
	`int MaxLength`	Returns the value of this facet.
	`string MinExclusive`	Returns the value of this facet.
	`string MinInclusive`	Returns the value of this facet.
	`int MinLength`	Returns the value of this facet.
	`string NamespaceURI`	Returns the namespace URI of the type.
	`string[] Patterns`	Returns the pattern facets, or null if no patterns are specified.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of this type.
	`int TotalDigits`	Returns the value of this facet.
	`WhitespaceType Whitespace`	Returns the whitespace normalization facet.

16.9.8 [YourSchema].[Doc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the members listed below. Note that, in the method names below, "Doc" stands for the name of the generated document class itself.

Methods

	Name	Description
	`static Doc CreateDocument()`	Creates a new, empty XML document.
	`static Doc CreateDocument(string encoding)`	Creates a new, empty XML document, with encoding of type "encoding".
	`static Doc`	Loads an XML document from a byte array.

	Name	Description
	`LoadFromBinary(byte[] binary)`	
	`static Doc LoadFromFile(string filename)`	Loads an XML document from a file.
	`static Doc LoadFromString(string xmlstring)`	Loads an XML document from a string.
	`byte[] SaveToBinary(bool prettyPrint)`	Saves an XML document to a byte array, with optional "pretty-print" formatting.
	`byte[] SaveToBinary(bool prettyPrint, string encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.
	`byte[] SaveToBinary(bool prettyPrint, string encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding, byte order, and BOM (Byte Order Mark).
	`void SaveToFile(string fileName, bool prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a file, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, string encoding, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, and line ending character(s).
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, bool bBigEndian, bool bBOM, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, byte order, BOM (Byte Order Mark), and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.

	Name	Description
	`void SaveToFileWithLineEnd(string fileName, bool prettyPrint, bool omitXmlDecl, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.
	`string SaveToString(bool prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
	`string SaveToString(bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a file, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SetDTDLocation(string dtdLocation)`	Adds a DOCTYPE declaration with the specified system ID. A root element must already exist.
	`void SetSchemaLocation(string schemaLocation)`	Adds an xsi:schemaLocation or xsi:noNamespaceSchemaLocation attribute to the root element. A root element must already exist.

16.9.9 [YourSchemaType].MemberAttribute

When code is generated from an XML schema, a class is created for each member attribute of a type. In the descriptions below, "AttributeType" stands for the type of the member attribute itself.

Methods

	Name	Description
	`bool Exists()`	Returns true if the attribute exists.
	`void Remove()`	Removes the attribute from its parent element.

Properties

	Name	Description
	`int EnumerationValue`	Generated for enumeration types only. Sets or gets the attribute value using one of the constants generated for the possible values. Returns **Invalid** if the value does not match any of the enumerated values in the schema.
	`Altova.Xml.Meta.AttributeInfo`	Returns an object for querying schema information (see `Altova.Xml.Meta.Attribute`).

	Name	Description
	`AttributeType Value`	Sets or gets the attribute value.

16.9.10 [YourSchemaType].MemberElement

When code is generated from an XML schema, a class with the following members is created for each member element of a type. The class implements the standard `System.Collections.IEnumerable` interface, so it can be used with the `foreach` statement.

In the descriptions below, "MemberType" stands for the type of the member element itself.

Methods

	Name	Description
	`MemberType Append()`	Creates a new element and appends it to its parent.
	`MemberType At(int index)`	Returns the member element specified by the index.
	`System.Collections.IEnumerator GetEnumerator()`	Returns an object for iterating instances of the member element.
	`void Remove()`	Deletes all occurrences of the element from its parent.
	`void RemoveAt(int index)`	Deletes the occurrence of the element specified by the index.

Properties

	Name	Description
	`int Count`	Returns the count of elements.
	`int EnumerationValue`	Generated for enumeration types only. Sets or gets the element value using one of the constants generated for the possible values. Returns **Invalid** if the value does not match any of the enumerated values in the schema.
	`bool Exists`	Returns true if at least one element exists.
	`MemberType First`	Returns the first instance of the member element.
	`Altova.Xml.Meta.ElementInfo`	Returns an object for querying schema information (see `Altova.Xml.Meta.Element`).
	`MemberType Last`	Returns the last instance of the member element.

	Name	Description
	`MemberType this[int index]`	Returns the member element specified by the index.
	`MemberType Value`	Sets or gets the element content (only generated if element can have mixed or simple content).

16.10 Reference to Generated Classes (Java)

This chapter includes a description of Java classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code may include other supporting classes, which are not listed here and are subject to modification.

16.10.1 com.altova.types.DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

	Name	Description
c	`public DateTime()`	Initializes a new instance of the `DateTime` class to an empty value.
c	`public DateTime(DateTime newvalue)`	Initializes a new instance of the `DateTime` class to the `DateTime` value supplied as argument.
c	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, int newoffsetTZ)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, the fractional part of the second, and timezone supplied as arguments. The fractional part of the second `newpartsecond` must be between 0 and 1. The timezone offset `newoffsetTZ` can be either positive or negative and is expressed in minutes.
c	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, and the fractional part of a second supplied as arguments.
c	`public DateTime(int newyear, int newmonth, int newday)`	Initializes a new instance of the `DateTime` class to the year, month, and day supplied as arguments.
c	`public DateTime(Calendar newvalue)`	Initializes a new instance of the `DateTime` class to the `java.util.Calendar` value supplied as argument.

Methods

	Name	Description
S	`static DateTime now()`	Returns the current time as a `DateTime` object.

S	`static DateTime parse(String s)`	Returns a `DateTime` object parsed from the string value supplied as argument. For example, the following sample string values would be converted successfully to a `DateTime` object:
		
		`2015-11-24T12:54:47.969+01:00`
		`2015-11-24T12:54:47`
		`2015-11-24`

	`int getDay()`	Returns the day of the current `DateTime` instance.
	`int getHour()`	Returns the hour of the current `DateTime` instance.
	`int getMillisecond()`	Returns the millisecond of the current `DateTime` instance, as an integer value.
	`int getMinute()`	Returns the minute of the current `DateTime` instance.
	`int getMonth()`	Returns the month of the current `DateTime` instance.
	`double getPartSecond()`	Returns the fractional part of the second of the current `DateTime` instance, as a `double` value. The return value is greater than zero and smaller than one, for example: `0.313`
	`int getSecond()`	Returns the second of the current `DateTime` instance.
	`int getTimezoneOffset()`	Returns the timezone offset, in minutes, of the current `DateTime` instance. For example, the timezone "UTC-01:00" would be returned as: `-60`
	`Calendar getValue()`	Returns the current `DateTime` instance as a `java.util.Calendar` value.
	`int getWeekday()`	Returns the day in week of the current `DateTime` instance. Values range from 0 through 6, where 0 is Monday (ISO-8601).
	`int getYear()`	Returns the year of the current `DateTime` instance.
	`int hasTimezone()`	Returns information about the timezone of the current `DateTime` instance. Possible return values are: `CalendarBase.TZ_MISSING` — A timezone offset is not defined. `CalendarBase.TZ_UTC` — The timezone is UTC. `CalendarBase.TZ_OFFSET` — A timezone offset has

	Name	Description
		been defined.
	`void setDay( int nDay )`	Sets the day of the current `DateTime` instance to the value supplied as argument.
	`void setHasTimezone( int nHasTZ )`	Sets the timezone information of the current `DateTime` instance to the value supplied as argument. This method can be used to strip the timezone information or set the timezone to UTC (Coordinated Universal Time). Valid values for the `nHasTZ` argument: `CalendarBase.TZ_MISSING` Set the timezone offset to undefined. `CalendarBase.TZ_UTC` Set the timezone to UTC. `CalendarBase.TZ_OFFSET` If the current object has a timezone offset, leave it unchanged.
	`void setHour( int nHour )`	Sets the hour of the current `DateTime` instance to the value supplied as argument.
	`void setMinute( int nMinute )`	Sets the minute of the current `DateTime` instance to the value supplied as argument.
	`void setMonth( int nMonth )`	Sets the month of the current `DateTime` instance to the value supplied as argument.
	`void setPartSecond( double nPartSecond )`	Sets the fractional part of the second of the current `DateTime` instance to the value supplied as argument.
	`void setSecond( int nSecond )`	Sets the second of the current `DateTime` instance to the value supplied as argument.
	`void setTimezoneOffset( int nOffsetTZ )`	Sets the timezone offset of the current `DateTime` instance to the value supplied as argument. The value `nOffsetTZ` must be an integer (positive or negative) and must be expressed in minutes.
	`void setYear( int nYear )`	Sets the year of the current `DateTime` instance to the value supplied as argument.
	`String toString()`	Returns the string representation of the current `DateTime` instance, for example: `2015-11-24T15:50:56.968+01:00`

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
import com.altova.types.*;
```

The following code listing illustrates various ways to create `DateTime` objects:

```
protected static void DateTimeExample1()
{
       // Initialize a new instance of the DateTime class to the current time
       DateTime dt = new DateTime(DateTime.now());
       System.out.println("DateTime created from current date and time:" +
dt.toString());

       // Initialize a new instance of the DateTime class by supplying the parts
       DateTime dt1 = new DateTime(2015, 11, 23, 14, 30, 24, .459);
       System.out.println("DateTime from parts (no timezone):" + dt1.toString());

       // Initialize a new instance of the DateTime class by supplying the parts
       DateTime dt2 = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
       System.out.println("DateTime from parts (with negative timezone):" +
dt2.toString());

       // Initialize a new instance of the DateTime class by parsing a string value
DateTime dt3 = DateTime.parse("2015-11-24T12:54:47.969+01:00");

       System.out.println("DateTime parsed from string:" + dt3.toString());

}
```

The following code listing illustrates getting values from `DateTime` objects:

```
protected static void DateTimeExample2()
   {
      // Initialize a new instance of the DateTime class to the current time
      DateTime dt = new DateTime(DateTime.now());

      // Output the formatted year, month, and day of this DateTime instance
      String str1 = String.format("Year: %d; Month: %d; Day: %d;",
dt.getYear(), dt.getMonth(), dt.getDay());
      System.out.println(str1);

      // Output the formatted hour, minute, and second of this DateTime
instance
      String str2 = String.format("Hour: %d; Minute: %d; Second: %d;",
dt.getHour(), dt.getMinute(), dt.getSecond());
      System.out.println(str2);

      // Return the timezone (in minutes) of this DateTime instance
      System.out.println("Timezone: " + dt.getTimezoneOffset());

      // Get the DateTime as a java.util.Calendar value
      java.util.Calendar dt_java = dt.getValue();
      System.out.println("" + dt_java.toString());

      // Return the day of week of this DateTime instance
      System.out.println("Weekday: " + dt.getWeekday());

      // Check whether the DateTime instance has a timezone defined
      switch(dt.hasTimezone())
      {
         case CalendarBase.TZ_MISSING:
            System.out.println("No timezone.");
```

```
            break;
         case CalendarBase.TZ_UTC:
            System.out.println("The timezone is UTC.");
            break;
         case CalendarBase.TZ_OFFSET:
            System.out.println("This object has a timezone.");
            break;
         default:
            System.out.println("Unable to determine whether a timezone is
defined.");
            break;
      }
   }
```

The following code listing illustrates changing the timezone offset of a `DateTime` object:

```
protected static void DateTimeExample3()
{
       // Create a new DateTime object with timezone -0100 UTC
       DateTime dt = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
       // Output the value before the change
       System.out.println("Before:" + dt.toString());
       // Change the offset to +0100 UTC
       dt.setTimezoneOffset(60);
       // Output the value after the change
       System.out.println("After: " + dt.toString());
}
```

16.10.2 com.altova.types.Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

	Name	Description
c	`Duration(Duration newvalue)`	Initializes a new instance of the `Duration` class to the `Duration` object supplied as argument.
c	`Duration(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, boolean newisnegative)`	Initializes a new instance of the `Duration` class to a duration built from parts supplied as arguments.

Methods

	Name	Description
s	`static Duration getFromDayTime( int`	Returns a `Duration` object created from the number of days, hours, minutes, seconds, and fractional

	Name	Description
	`newday, int newhour, int newminute, int newsecond, double newpartsecond )`	second parts supplied as argument.
S	`static Duration getFromYearMonth( int newyear, int newmonth )`	Returns a `Duration` object created from the number of years and months supplied as argument.
S	`static Duration parse( String s )`	Returns a `Duration` object created from the string supplied as argument. For example, the string `-P1Y1M1DT1H1M1.333S` can be used to create a negative duration of one year, one month, one day, one hour, one minute, one second, and 0.333 fractional parts of a second. To create a negative duration, append the minus sign (-) to the string.
S	`static Duration parse( String s, ParseType pt )`	Returns a `Duration` object created from the string supplied as argument, using a specific parse format. The parse format can be any of the following: **ParseType.DAYTIME** — Must be used when the string **s** consists of any of the following: days, hours, minutes, seconds, fractional second parts, for example `-P4DT4H4M4.774S`. **ParseType.DURATION** — Must be used when the string **s** consists of any of the following: years, months, days, hours, minutes, seconds, fractional second parts, for example `P1Y1M1DT1H1M1.333S`. **ParseType.YEARMONTH** — Must be used when the string **s** consists of any of the following: years, months. For example: `P3Y2M`.
	`int getDay()`	Returns the number of days in the current `Duration` instance.
	`long getDayTimeValue()`	Returns the day and time value (in milliseconds) of the current `Duration` instance. Years and months are ignored.
	`int getHour()`	Returns the number of hours in the current `Duration` instance.
	`int getMillisecond()`	Returns the number of milliseconds in the current `Duration` instance.
	`int getMinute()`	Returns the number of minutes in the current `Duration` instance.
	`int getMonth()`	Returns the number of months in the current

	Name	Description
		`Duration` instance.
●	`double getPartSecond()`	Returns the number of fractional second parts in the current `Duration` instance.
●	`int getSecond()`	Returns the number of seconds in the current `Duration` instance.
●	`int getYear()`	Returns the number of years in the current `Duration` instance.
●	`int getYearMonthValue()`	Returns the year and month value (in months) of the current `Duration` instance. Days, hours, seconds, and milliseconds are ignored.
●	`boolean isNegative()`	Returns Boolean **true** if the current `Duration` instance is positive.
●	`void setDayTimeValue(long l)`	Sets the duration to the number of milliseconds supplied as argument, affecting only the day and time part of the duration.
●	`void setNegative( boolean isnegative )`	Converts the current `Duration` instance to a negative duration.
●	`void setYearMonthValue(int l)`	Sets the duration to the number of months supplied as argument. Only the years and months part of the duration is affected.
●	`String toString()`	Returns the string representation of the current `Duration` instance, for example: `-P4DT4H4M4.774S`
●	`String toYearMonthString()`	Returns the string representation of the YearMonth part of the current `Duration` instance, for example: `P1Y2M`

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
import com.altova.types.*;
import com.altova.types.Duration.ParseType;
```

The following code listing illustrates various ways to create `Duration` objects:

```
protected static void ExampleDuration()
{
        // Create a negative duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1 second,
        // and 0.333 fractional second parts
        Duration dr = new Duration(1, 1, 1, 1, 1, 1, .333, true);

        // Create a duration from an existing Duration object
```

```
        Duration dr1 = new Duration(dr);

        // Create a duration of 4 days, 4 hours, 4 minutes, 4 seconds, 774 fractionalsecond parts
        Duration dr2 = Duration.getFromDayTime(4, 4, 4, 4, .774);

        // Create a duration of 3 years and 2 months
        Duration dr3 = Duration.getFromYearMonth(3, 2);

        // Create a duration from a string
        Duration dr4 = Duration.parse("-P4DT4H4M4.774S");

        // Create a duration from a string, using specific parse formats
        Duration dr5 = Duration.parse("-P1Y1M1DT1H1M1.333S",
ParseType.DURATION);
        Duration dr6 = Duration.parse("P3Y2M", ParseType.YEARMONTH);
        Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

The following code listing illustrates getting and setting the value of `Duration` objects:

```
protected static void DurationExample2()
{
        // Create a duration of 1 year, 2 month, 3 days, 4 hours, 5 minutes, 6 seconds,
    // and 333 milliseconds
        Duration dr = new Duration(1, 2, 3, 4, 5, 6, .333, false);
        // Output the number of days in this duration
        System.out.println(dr.getDay());

        // Create a positive duration of one year and 333 milliseconds
        Duration dr1 = new Duration(1, 0, 0, 0, 0, 0, .333, false);
        // Output the day and time value in milliseconds
        System.out.println(dr1.getDayTimeValue());

        // Create a positive duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1 second,
    // and 333 milliseconds
        Duration dr2 = new Duration(1, 1, 1, 1, 1, 1, .333, false);
        // Output the year and month value in months
        System.out.println(dr2.getYearMonthValue());

        // Create a positive duration of 1 year and 1 month
        Duration dr3 = new Duration(1, 1, 0, 0, 0, 0, 0, false);
        // Output the value
        System.out.println("The duration is now:" + dr3.toString());
        // Set the DayTime part of duration to 1000 milliseconds
        dr3.setDayTimeValue(1000);
        // Output the value
        System.out.println("The duration is now:" + dr3.toString());
        // Set the YearMonth part of duration to 1 month
        dr3.setYearMonthValue(1);
        // Output the value
        System.out.println("The duration is now:" + dr3.toString());
        // Output the year and month part of the duration
        System.out.println("The YearMonth part of the duration is:" +
dr3.toYearMonthString());
}
```

16.10.3 com.altova.xml.meta.Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

Methods

	Name	Description
	`SimpleType getDataType()`	Returns the type of the attribute content.
	`String getLocalName()`	Returns the local name of the attribute.
	`String getNamespaceURI()`	Returns the namespace URI of the attribute.
	`boolean isRequired()`	Returns true if the attribute is required.

16.10.4 com.altova.xml.meta.ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Methods

	Name	Description
	`Attribute findAttribute(String localName, String namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
	`Element findElement(String localName, String namespaceURI)`	Finds the element with the specified local name and namespace URI.
	`Attribute[] GetAttributes()`	Returns a list of all attributes.
	`ComplexType getBaseType()`	Returns the base type of this type.
	`SimpleType getContentType()`	Returns the simple type of the content.
	`Element[] GetElements()`	Returns a list of all elements.
	`String getLocalName()`	Returns the local name of the type.

	Name	Description
	`String getNamespaceURI()`	Returns the namespace URI of the type.

16.10.5 com.altova.xml.meta.Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

Methods

	Name	Description
	`ComplexType getDataType()`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use `getContentType()` of the returned object to get the simple content type.
	`String getLocalName()`	Returns the local name of the element.
	`int getMaxOccurs()`	Returns the maxOccurs value defined in the schema.
	`int getMinOccurs()`	Returns the minOccurs value defined in the schema.
	`String getNamespaceURI()`	Returns the namespace URI of the element.

16.10.6 com.altova.xml.meta.SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Methods

	Name	Description
	`SimpleType getBaseType()`	Returns the base type of this type.
	`String[] getEnumerations()`	Returns an array of all enumeration facets.
	`int getFractionDigits()`	Returns the value of this facet.
	`int getLength()`	Returns the value of this facet.
	`String getLocalName()`	Returns the local name of the type.

	Name	Description
	`String getMaxExclusive()`	Returns the value of this facet.
	`String getMaxInclusive()`	Returns the value of this facet.
	`int getMaxLength()`	Returns the value of this facet.
	`String getMinExclusive()`	Returns the value of this facet.
	`String getMinInclusive()`	Returns the value of this facet.
	`int getMinLength()`	Returns the value of this facet.
	`String getNamespaceURI()`	Returns the namespace URI of the type.
	`String[] getPatterns()`	Returns an array of all pattern facets.
	`int getTotalDigits()`	Returns the value of this facet.
	`int getWhitespace()`	Returns the value of the whitespace facet, which is one of: `com.altova.typeinfo.WhitespaceType.Whitespace_Unknown` `com.altova.typeinfo.WhitespaceType.Whitespace_Preserve` `com.altova.typeinfo.WhitespaceType.Whitespace_Replace` `com.altova.typeinfo.WhitespaceType.Whitespace_Collapse`

16.10.7 com.[YourSchema].[Doc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the members listed below. Note that, in the method names below, "Doc" stands for the name of the generated document class itself.

Methods

	Name	Description
S	`static Doc createDocument()`	Creates a new, empty XML document.
S	`static Doc loadFromBinary(byte[] xml)`	Loads an XML document from a byte array.
S	`static Doc loadFromFile(String fileName)`	Loads an XML document from a file.
S	`static Doc loadFromString(String`	Loads an XML document from a string.

	Name	Description
	`xml)`	
●	`byte[] saveToBinary(boolean prettyPrint)`	Saves an XML document to a byte array, with optional "pretty-print" formatting.
●	`byte[] saveToBinary(boolean prettyPrint, String encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.
●	`byte[] saveToBinary(boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
●	`void saveToFile(String fileName, boolean prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
●	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with UTF-8 encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
●	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
●	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding, boolean bBigEndian, boolean bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
●	`void saveToFile(String fileName, boolean prettyPrint, String encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding.
●	`void saveToFile(String fileName, boolean prettyPrint, String encoding, boolean bBigEndian, boolean bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
●	`String saveToString(boolean prettyPrint)`	Saves an XML document to a string, with optional "pretty-print" formatting.
●	`String saveToString(boolean prettyPrint, boolean omitXmlDecl)`	Saves an XML document to a string, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.

	Name	Description
	`void setSchemaLocation(String schemaLocation)`	Adds an `xsi:schemaLocation` or `xsi:noNamespaceSchemaLocation` attribute to the root element. A root element must already exist.

16.10.8 com.[YourSchema].[YourSchemaType].MemberAttribute

When code is generated from an XML schema, a class is created for each member attribute of a type. In the descriptions below, "AttributeType" stands for the type of the member attribute itself.

Methods

	Name	Description
	`boolean exists()`	Returns true if the attribute exists.
	`int getEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
	`com.altova.xml.meta.Attribute getInfo()`	Returns an object for querying schema information (see `com.altova.xml.meta.Attribute`).
	`AttributeType getValue()`	Gets the attribute value.
	`void remove()`	Removes the attribute from its parent element.
	`void setEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.
	`void setValue(AttributeType value)`	Sets the attribute value.

16.10.9 com.[YourSchema].[YourSchemaType].MemberElement

When code is generated from an XML schema, a class with the following members is created for each member element of a type. In the descriptions below, "MemberType" stands for the type of the member element itself.

Methods

	Name	Description
	`MemberType append()`	Creates a new element and appends it to its

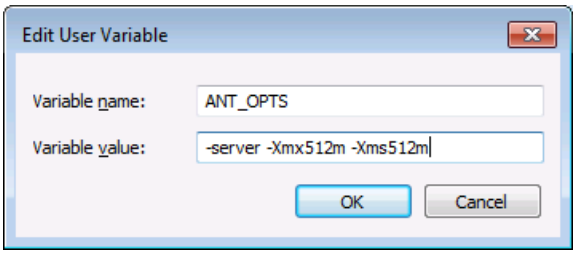
	Name	Description
		parent.
●	`MemberType at(int index)`	Returns the instance of the member element at the specified index.
●	`int count()`	Returns the count of elements.
●	`boolean exists()`	Returns true if at least one element exists.
●	`MemberType first()`	Returns the first instance of the member element.
●	`int getEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
●	`com.altova.xml.meta.Element getInfo()`	Returns an object for querying schema information (see `com.altova.xml.meta.Element`).
●	`MemberType getValue()`	Gets the element content (only generated if element can have simple or mixed content).
●	`java.util.Iterator iterator()`	Returns an object for iterating instances of the member element.
●	`MemberType last()`	Returns the last instance of the member element.
●	`void remove()`	Deletes all occurrences of the element from its parent.
●	`void removeAt(int index)`	Deletes the occurrence of the element specified by the index.
●	`void setEnumerationValue(int index)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.
●	`void setValue(MemberType value)`	Sets the element content (only generated if element can have simple or mixed content).

16.11 Code Generation Tips

Resolving "Out of memory" exceptions during Java compilation

Complex mappings with large schemas can produce a large amount of code, which might cause a java.lang.OutofMemory exception during compilation using Ant. To rectify this:

- Add the environment variable `ANT_OPTS`, which sets specific Ant options such as the memory to be allocated to the compiler, and add values as shown below.



- To make sure that the compiler and the generated code run in the same process as Ant, change the `fork` attribute, in **build.xml**, to `false`.

You may need to customize the values depending on the amount of memory in your machine and the size of the project you are working with. For more details, see your Java VM documentation.

When running the `ant jar` command, you may get an error message similar to "`[...] archive contains more than 65535 entities`". To prevent this, it is recommended that you use Ant 1.9 or later, and, in the **build.xml** file, add `zip64mode="as-needed"` to the `<jar>` element.

Reserving method names

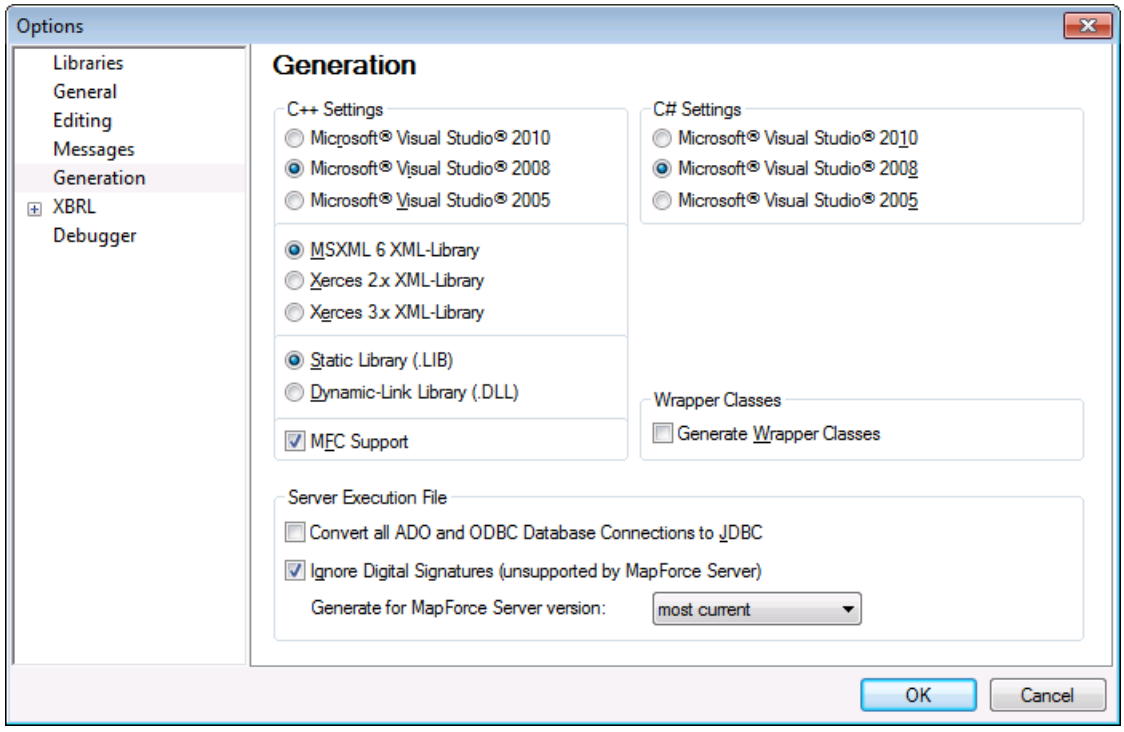
When customizing code generation using the supplied SPL files, it might be necessary to reserve names to avoid collisions with other symbols. To do this:

1. Navigate to subdirectory corresponding to the programming language of the **spl** subdirectory of the program installation directory e.g. `C:\Program Files\Altova\MapForce2017\spl\java\`.
2. Open the **settings.spl** file and insert a new line into the reserve section e.g. reserve "myReservedWord".
3. Regenerate the program code.

16.12 Code Generator Options

To view or change the MapForce settings applicable to code generation:

- On the **Tools** menu, click **Options**, and then click **Generation**.



The available settings are as follows.

C++ Settings	Defines the specific compiler settings for the C++ environment, namely: • The Visual Studio edition (2005, 2008, 2010) • The XML library (MSXML, Xerces 2.x, Xerces 3.x) • Whether static or dynamic libraries must be generated • Whether code must be generated with or without MFC support
C# Settings	Defines the specific compiler settings for the C# environment, namely, the Visual Studio edition (2005, 2008, 2010).
Wrapper Classes	Allows you to generate wrapper classes for XML schemas. These wrapper classes can be used by custom code that includes the code generated by MapForce.

Options grouped under "Server Execution File" are applicable when you compile mappings to MapForce Server execution files. They do not affect generation of C#, C++, or Java code. For more information, see Compiling Mappings to MapForce Server Execution Files.

16.13 SPL (Spy Programming Language)

This section gives an overview of Spy Programming Language, the code generator's template language.

It is assumed that you have prior programming experience, and are familiar with operators, functions, variables and classes, as well as the basics of object-oriented programming - which is used heavily in SPL.

The templates used by MapForce are supplied in the ...\ MapForce\ spl folder. You can use these files as an aid to help you in developing your own templates.

How code generator works

Inputs to the code generator are the template files (.spl) and the object model provided by MapForce. The template files contain SPL instructions for creating files, reading information from the object model and performing calculations, interspersed with literal code fragments in the target programming language.

The template file is interpreted by the code generator and outputs **.cpp, .java, .cs** source code files, project files, or any other type of file depending on the template. The source code can then be compiled into an executable file that accesses XML data described by the schema file.

SPL files have access to a wide variety of information that is collated from the source schemas. Please note that an SPL file is not tied to a specific schema, but allows access to all schemas! Make sure you write your SPL files generically, avoid structures etc. which apply to specific schemas!

Example: Creating a new file in SPL:

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

This is a very basic SPL file. It creates a file named **test.cpp,** and places the include statement within it. The close command completes the template.

16.13.1 Basic SPL structure

An SPL file contains literal text to output, interspersed with code generator instructions.

Code generator instructions are enclosed in square brackets '**[**' and '**]**'.
Multiple statements can be included in a bracket pair. Additional statements have to be separated by a new line or a colon '**:**'.

Valid examples are:

```
[$x = 42
$x = $x + 1]
```

or

```
[$x = 42: $x = $x + 1]
```

Adding text to files

Text not enclosed by **[** and **]**, is written directly to the current output file. If there is no current output file, the text is ignored (see Using files how to create an output file).
To output literal square brackets, escape them with a backslash: **\[** and **\]**; to output a backslash use ****.

Comments

Comments inside an instruction block always begin with a **'** character, and terminate on the next line, or at a block close character **]**.

16.13.2 Declarations

The following statements are evaluated while parsing the SPL template file. They are **not** affected by flow control statements like conditions, loops or subroutines, and are always evaluated exactly once.

These keywords, like all keywords in SPL, are not case sensitive.

Remember that all of these declarations must be inside a block delimited by square brackets.

map *mapname key* **to** *value* [, *key* **to** *value*]...

This statement adds information to a map. See below for specific uses.

map schemanativetype *schematype* **to** *typespec*

The specified built-in XML Schema type will be mapped to the specified native type or class, using the specified formatter. This setting applies only to code generation for version 2007r3 and higher. Typespec is a native type or class name, followed by a comma, followed by the formatter class instance.

Example:
```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

map type *schematype* **to** *classname*

The specified built-in XML Schema type will be mapped to the specified class. This setting applies only to code generation for version 2007 or lower.

Example:
```
map type "float" to "CSchemaFloat"
```

default *setting* **is** *value*

This statement allows you to affect how class and member names are derived from the XML Schema.
Note that the setting names are case sensitive.

Example:
```
default "InvalidCharReplacement" is "_"
```

Setting name	Explanation
ValidFirstCharSet	Allowed characters for starting an identifier

Setting name	Explanation
ValidCharSet	Allowed characters for other characters in an identifier
InvalidCharReplacement	The character that will replace all characters in names that are not in the ValidCharSet
AnonTypePrefix	Prefix for names of anonymous types*
AnonTypeSuffix	Suffix for names of anonymous types*
ClassNamePrefix	Prefix for generated class names
ClassNameSuffix	Suffix for generated class names
EnumerationPrefix	Prefix for symbolic constants declared for enumeration values
EnumerationUpperCase	"on" to convert the enumeration constant names to upper case
FallbackName	If a name consists only of characters that are not in ValidCharSet, use this one

* Names of anonymous types are built from AnonTypePrefix + element name + AnonTypeSuffix

reserve *word*

Adds the specified word to the list of reserved words. This ensures that it will never be generated as a class or member name.

Example:
```
reserve "while"
```

include *filename*

Example:
```
include "Module.cpp"
```

includes the specified file as SPL source. This allows you to split your template into multiple files for easier editing and handling.

16.13.3 Variables

Any non-trivial SPL file will require variables. Some variables are predefined by the code generator, and new variables may be created simply by assigning values to them.

The **$** character is used when **declaring** or **using** a variable, a variable name is always prefixed by **$**.
Variable names are **case sensitive**.

Variables types:

- integer - also used as boolean, where 0 is false and everything else is true
- string
- object - provided by MapForce
- iterator - see foreach statement

Variable types are declared by first assignment:

```
[$x = 0]
```
x is now an integer.

```
[$x = "teststring"]
```
x is now treated as a string.

Strings

String constants are always enclosed in double quotes, like in the example above. **\n** and **\t** inside double quotes are interpreted as newline and tab, **\"** is a literal double quote, and **** is a backslash. String constants can also span multiple lines.

String concatenation uses the **&** character:

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

Objects

Objects represent the information contained in the XML schemas, database structures, text files and mappings. Objects have **properties**, which can be accessed using the **.** operator. It is not possible to create new objects in SPL (they are predefined by the code generator, derived from the input mapping), but it is possible to assign objects to variables.

Example:

```
class [=$class.Name]
```

This example outputs the word "class", followed by a space and the value of the **Name** property of the **$class** object.

16.13.4 Predefined variables

After a Schema file is analyzed by the code generator, the objects in the table below exist in the Template Engine.

Name	Type	Description
$schematype	integer	1 for DTD, 2 for XML Schema
$TheLibrary	Library	The library derived from the XML Schema or DTD
$module	string	Name of the source Schema without extension
$outputpath	string	The output path specified by the user, or the default output path

For **C++** generation only:

Name	Type	Description
$domtype	integer	1 for MSXML, 2 for Xerces

Name	Type	Description
$XercesVersion	integer	2 for Xerces 2.x, 3 for Xerces 3.x
$libtype	integer	1 for static LIB, 2 for DLL
$mfc	boolean	True if MFC support is enabled
$vc6project	boolean	True if Visual C++ project files are to be generated
$VS2005Project	boolean	True if Visual C++ 2005 project files are to be generated
$VS2008Project	boolean	True if Visual C++ 2008 project files are to be generated
$VS2010Project	boolean	True if Visual C++ 2010 project files are to be generated

For **C#** generation only:

Name	Type	Description
$CreateVS2005Project	boolean	True if Visual Studio 2005 project files are to be generated
$CreateVS2008Project	boolean	True if Visual Studio 2008 project files are to be generated
$CreateVS2010Project	boolean	True if Visual Studio 2010 project files are to be generated

16.13.5 Creating output files

These statements are used to create output files from the code generation.

Remember that all of these statements must be inside a block delimited by square brackets.

create *filename*

creates a new file. The file has to be closed with the **close** statement. All following output is written to the specified file.

Example:

```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name &
".java"]
package [=$JavaPackageName];

public class [=$application.Name]Application {
...
}
[close]
```

close

closes the current output file.

=$*variable*

writes the value of the specified variable to the current output file.

Example:
```
[$x = 20+3]
The result of your calculation is [=$x] - so have a nice day!
```

- The file output will be:

```
The result of your calculation is 23 - so have a nice day!
```

write *string*

writes the string to the current output file.

Example:
```
[write "C" & $name]
```

This can also be written as:
```
C[=$name]
```

filecopy *source* **to** *target*

copies the source file to the target file, without any interpretation.

Example:
```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

16.13.6 Operators

Operators in SPL work like in most other programming languages.

List of SPL operators in descending precedence order:

.	Access object property
( )	Expression grouping
true	boolean constant "true"
false	boolean constant "false"
&	String concatenation
-	Sign for negative number
not	Logical negation
*	Multiply
/	Divide
%	Modulo
+	Add
-	Subtract
<=	Less than or equal
<	Less than

>= Greater than or equal
> Greater than

= Equal
<> Not equal

and Logical conjunction (with short circuit evaluation)
or Logical disjunction (with short circuit evaluation)

= Assignment

16.13.7 Conditions

SPL allows you to use standard "if" statements. The syntax is as follows:

```
if condition
       statements
else
       statements
endif
```

or, without else:

```
if condition
       statements
endif
```

Please note that there are no round brackets enclosing the condition!
As in any other programming language, conditions are constructed with logical and comparison operators.

Example:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
       whatever you want ['inserts whatever you want, in the resulting file]
[endif]
```

Switch

SPL also contains a multiple choice statement.

Syntax:

```
switch $variable
       case X:
              statements
       case Y:
       case Z:
              statements
       default:
              statements
endswitch
```

The case labels must be constants or variables.

The switch statement in SPL does not fall through the cases (as in C), so there is no need for a "break" statement.

16.13.8 Collections and foreach

Collections and iterators

A collection contains multiple objects - like a ordinary array. Iterators solve the problem of storing and incrementing array indexes when accessing objects.

Syntax:

```
foreach iterator in collection
      statements
next
```

Example:

```
[foreach $class in $classes
      if not $class.IsInternal
            ]     class [=$class.Name];
[     endif
next]
```

Example 2:

```
[foreach $i in 1 To 3
     Write "// Step " & $i & "\n"
     ' Do some work
next]
```

The first line:
$classes is the **global object** of all generated types. It is a collection of single class objects.

Foreach steps through all the items in $classes, and executes the code following the instruction, up to the **next** statement, for each of them.

In each iteration, **$class** is assigned to the next class object. You simply work with the class object instead of using, classes[i]->Name(), as you would in C++.

All collection iterators have the following additional properties:

Index	The current index, starting with 0
IsFirst	true if the current object is the first of the collection (index is 0)
IsLast	true if the current object is the last of the collection
Current	The current object (this is implicit if not specified and can be left out)

Example:

```
[foreach $enum in $facet.Enumeration
      if not $enum.IsFirst
            ], [
      endif
      ]"[=$enum.Value]"[
next]
```

16.13.9 Subroutines

Code generator supports subroutines in the form of procedures or functions.

Features:

- By-value and by-reference passing of values
- Local/global parameters (local within subroutines)
- Local variables
- Recursive invocation (subroutines may call themselves)

16.13.9.1 Subroutine declaration

Subroutines

Syntax example:

```
Sub SimpleSub()
...lines of code
EndSub
```

- **Sub** is the keyword that denotes the procedure.
- **SimpleSub** is the name assigned to the subroutine.
- Round **parenthesis** can contain a parameter list.
- The code block of a subroutine starts immediately after the closing parameter parenthesis.
- **EndSub** denotes the end of the code block.

Please note:
Recursive or cascaded subroutine **declaration** is not permitted, i.e. a subroutine may not contain another subroutine.

Parameters
Parameters can also be passed by procedures using the following syntax:

- All parameters must be variables
- Variables must be prefixed by the **$** character
- Local variables are defined in a subroutine
- Global variables are declared explicitly, outside of subroutines
- Multiple parameters are separated by the comma character "**,**" within round parentheses
- Parameters can pass values

Parameters - passing values
Parameters can be passed in two ways, by value and by reference, using the keywords **ByVal** and **ByRef** respectively.

Syntax:

```
'define sub CompleteSub()
[Sub CompleteSub($param,ByVal$paramByValue,ByRef$paramByRef)
]...
```

- **ByVal** specifies that the parameter is passed by value. Note that most objects can only be

passed by reference.

- **ByRef** specifies that the parameter is passed by reference. This is the default if neither ByVal nor ByRef is specified.

Function return values
To return a value from a subroutine, use the **return** statement. Such a function can be called from within an expression.

Example:

```
'define a function
[Sub MakeQualifiedName( ByVal$namespacePrefix,ByVal$localName )
if$namespacePrefix = ""
  return $localName
else
  return $namespacePrefix & ":" & $localName
endif
EndSub
]
```

16.13.9.2 Subroutine invocation

Use **call** to invoke a subroutine, followed by the procedure name and parameters, if any.

```
CallSimpleSub()
```

or,

```
CallCompleteSub( "FirstParameter",$ParamByValue,$ParamByRef )
```

Function invocation

To invoke a function (any subroutine that contains a **return** statement), simply use its name inside an expression. Do not use the **call** statement to call functions.
Example:

```
$QName = MakeQualifiedName($namespace,"entry")
```

16.13.9.3 Subroutine example

Highlighted example showing subroutine declaration and invocation.

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
]
```

The same sample code:

```
[create $outputpath & $module & "output.txt"

'define sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

'execute sub SimpleSub()
CallSimpleSub()

$ParamByValue       = "OriginalValue"
]ParamByValue       = [=$ParamByValue]
[$ParamByRef = "OriginalValue"
]ParamByRef  = [=$ParamByRef]

'define sub CompleteSub()
[Sub CompleteSub( $param,ByVal$paramByValue,ByRef$paramByRef )
]CompleteSub called.
        param = [=$param]
        paramByValue = [=$paramByValue]
        paramByRef = [=$paramByRef]
[$ParamByRef = "LocalVariable"
```

```
$paramByValue = "new value"
$paramByRef = "new value"
]      Set values inside Sub
[$paramByRef = "LocalVariable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

'run sub CompleteSub()
CallCompleteSub("FirstParameter",$paramByValue,$paramByRef)
]
ParamByValue=[=$paramByValue]
ParamByRef=[=$paramByRef]
[
Close
]
```

16.13.10 Built in Types

The section describes the properties of the built-in types used in the predefined variables which describe the parsed schema.

16.13.10.1 Library

This object represents the whole library generated from the XML Schema or DTD.

Property	Type	Description
SchemaNamespaces	Namespace collection	Namespaces in this library
SchemaFilename	string	Name of the XSD or DTD file this library is derived from
SchemaType	integer	1 for DTD, 2 for XML Schema
Guid	string	A globally unique ID
CodeName	string	Generated library name (derived from schema file name)

16.13.10.2 Namespace

One namespace object per XML Schema namespace is generated. Schema components that are not in any namespace are contained in a special namespace object with an empty NamespaceURI.
Note that for DTD, namespaces are also derived from attributes whose names begin with "xmlns".

Property	Type	Description
CodeName	string	Name for generated code (derived from prefix)
LocalName	string	Namespace prefix
NamespaceURI	string	Namespace URI

Property	Type	Description
Types	Type collection	All types contained in this namespace
Library	Library	Library containing this namespace

16.13.10.3 Type

This object represents a complex or simple type. It is used to generate a class in the target language.
There is one additional type per library that represents the document, which has all possible root elements as members.
Anonymous types have an empty LocalName.

Property	Type	Description
CodeName	string	Name for generated code (derived from local name or parent declaration)
LocalName	string	Original name in the schema
Namespace	Namespace	Namespace containing this type
Attributes	Member collection	Attributes contained in this type*
Elements	Member collection	Child elements contained in this type
IsSimpleType	boolean	True for simple types, false for complex types
IsDerived	boolean	True if this type is derived from another type, which is also represented by a Type object
IsDerivedByExtension	boolean	True if this type is derived by extension
IsDerivedByRestriction	boolean	True if this type is derived by restriction
IsDerivedByUnion	boolean	True if this type is derived by union
IsDerivedByList	boolean	True if this type is derived by list
BaseType	Type	The base type of this type (if IsDerived is true)
IsDocumentRootType	boolean	True if this type represents the document itself
Library	Library	Library containing this type
IsFinal	boolean	True if declared as final in the schema
IsMixed	boolean	True if this type can have mixed content
IsAbstract	boolean	True if this type is declared as abstract

Property	Type	Description
IsGlobal	boolean	True if this type is declared globally in the schema
IsAnonymous	boolean	True if this type is declared locally in an element

For simple types only:

Property	Type	Description
IsNativeBound	boolean	True if native type binding exists
NativeBinding	NativeBinding	Native binding for this type
Facets	Facets	Facets of this type
Whitespace	string	Shortcut to the Whitespace facet

* Complex types with text content (these are types with mixed content and complexType with simpleContent) have an additional unnamed attribute member that represents the text content.

16.13.10.4 Member

This object represents an attribute or element in the XML Schema. It is used to create class members of types.

Property	Type	Description
CodeName	string	Name for generated code (derived from local name or parent declaration)
LocalName	string	Original name in the schema. Empty for the special member representing text content of complex types.
NamespaceURI	string	The namespace URI of this Element/ Attribute within XML instance documents/ streams.
DeclaringType	Type	Type originally declaring the member (equal to ContainingType for non-inherited members)
ContainingType	Type	Type where this is a member of
DataType	Type	Data type of this member's content
Library	Library	Library containing this member's DataType
IsAttribute	boolean	True for attributes, false for elements
IsOptional	boolean	True if minOccurs = 0 or optional attribute
IsRequired	boolean	True if minOccurs > 0 or required

Property	Type	Description
		attribute
IsFixed	boolean	True for fixed attributes, value is in Default property
IsDefault	boolean	True for attributes with default value, value is in Default property
IsNillable	boolean	True for nillable elements
IsUseQualified	boolean	True if NamespaceURI is not empty
MinOccurs	integer	minOccurs, as in schema. 1 for required attributes
MaxOccurs	integer	maxOccurs, as in schema. 0 for prohibited attributes, - 1 for unbounded
Default	string	Default value

16.13.10.5 NativeBinding

This object represents the binding of a simple type to a native type in the target programming language, as specified by the "schemanativetype" map.

Property	Type	Description
ValueType	string	Native type
ValueHandler	string	Formatter class instance

16.13.10.6 Facets

This object represents all facets of a simple type.
Inherited facets are merged with the explicitly declared facets. If a Length facet is in effect, MinLength and MaxLength are set to the same value.

Property	Type	Description
DeclaringType	Type	Type facets are declared on
Whitespace	string	"preserve", "collapse" or "replace"
MinLength	integer	Facet value
MaxLength	integer	Facet value
MinInclusive	integer	Facet value
MinExclusive	integer	Facet value
MaxInclusive	integer	Facet value
MaxExclusive	integer	Facet value
TotalDigits	integer	Facet value

Property	Type	Description
FractionDigits	integer	Facet value
List	Facet collection	All facets as list

Facet

This object represents a single facet with its computed value effective for a specific type.

Property	Type	Description
LocalName	string	Facet name
NamespaceURI	string	Facet namespace
FacetType	string	one of "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" or "valuespace-range"
DeclaringType	Type	Type this facet is declared on
FacetCheckerName	string	Name of facet checker (from schemafacet map)
FacetValue	string or integer	Actual value of this facet

Chapter 17

The MapForce API

17 The MapForce API

The COM-based API of MapForce enables clients to access the functionality of MapForce from a custom code or application, and automate a wide range of tasks.

The MapForce COM API follows the common specifications for automation servers as set out by Microsoft. MapForce is automatically registered as a COM server object during installation. Once the COM server object is registered, you can invoke it from within applications and scripting languages that have programming support for COM calls. This makes it possible to access the MapForce API not only from development environments using .NET, C++ and Visual Basic, but also from scripting languages like JScript and VBScript.

Note: If you use the MapForce API to create an application that you intend to distribute to other clients, MapForce must be installed on each client computer. Also, your custom integration code must be deployed to (or your application installed on) each client computer.

(C) 2017 Altova GmbH

17.1 Overview

This overview of the MapForce API provides you with the object model for the API and a description of the most important API concepts. The following topics are covered:

- Accessing the API
- The Object Model
- Error Handling
- Examples

17.1.1 Accessing the API

To access the MapForce COM API, a new instance of the `Application` object must be created in your application (or script). Once this object is created, you can interact with MapForce by invoking its methods and properties as required (for example, create a new document, open an existing document, generate mapping code, etc).

Prerequisites

To make the MapForce COM object available in your Visual Studio project, add a reference to the MapForce type library (.tlb) file. The following instructions are applicable to Visual Studio 2013, but are similar in other Visual Studio versions:

1. On the **Project** menu, click **Add Reference**.
2. Click **Browse** and select the **MapForce.tlb** file located in the MapForce installation folder.

A sample MapForce API client in C# is available at: **(My) Documents\Altova\MapForce2017 \MapForceExamples\API\C#**.

In Java, the MapForce API is available through Java-COM bridge libraries. These libraries are available in the MapForce installation folder: **C:\Program Files (x86)\Altova\MapForce2017 \JavaAPI** (note this path is valid when 32-bit MapForce runs on 64-bit Windows, otherwise adjust the path accordingly).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceAPI.jar`: Java classes that wrap the MapForce automation interface
- `MapForceAPI_JavaDoc.zip`: a `Javadoc` file containing help documentation for the Java API

To allow access to the MapForce automation server directly from Java code, the libraries above must be in the Java `classpath`.

A sample MapForce API client in Java is available at: **(My) Documents\Altova\MapForce2017 \MapForceExamples\API\Java**.

In scripting languages such as JScript or VBScript, the MapForce COM object is accessible through the Microsoft Windows Script Host (see https://msdn.microsoft.com/en-us/library/9bbdkx3k.aspx). Such scripts can be written with a text editor, and do not need compilation, since they are executed by the Windows Script Host packaged with Windows. (To check that the Windows Script Host is running, type `wscript.exe /?` at the command prompt). A sample MapForce API client in JScript is available at: **(My) Documents\Altova**

\MapForce2017\MapForceExamples\API\JScript.

Guidelines

The following guidelines should be considered in your client code:

- Do not hold references to objects in memory longer than you need them. If a user interacts between two calls of your client, then there is no guarantee that these references are still valid.
- Be aware that if your client code crashes, instances of MapForce may still remain in the system. For details on how to avoid error messages, see Error handling.
- Free references explicitly, if using languages such as C++.

Creating the Application object

The syntax to create the starting `Application` object depends on the programming language, as shown in the examples below:

Visual Basic

```
Dim objMapForce As MapForceLib.Application = New MapForceLib.Application
```

VBA

```
' Create a new instance of MapForce.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
```

VBScript

```
' Access a running instance, or create a new instance of MapForce.
Set objMapForce = GetObject("MapForce.Application");
```

C#

```
// Create a new instance of MapForce via its automation interface.
MapForceLib.Application objMapForce = new MapForceLib.Application();
```

Java

```
// Start MapForce as COM server.
com.altova.automation.MapForce.Application objMapForce = new Application();
// COM servers start up invisible so we make it visible
objMapForce.setVisible(true);
```

JScript

```
// Access a running instance, or create a new instance of MapForce.
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");}
```

17.1.2 The Object Model

The starting point for every application which uses the MapForce API is the `Application` object. All other interfaces are accessed through the `Application` object as the starting point.

The object model of the MapForce API can be represented as follows (each indentation level indicates a child–parent relationship with the level directly above):

```
Application
    Options
    Project
        ProjectItem
    Documents
        Document
            MapForceView
            Mapping
                Component
                    Datapoint
                Components
                Connection
            Mappings
    ErrorMarkers
        ErrorMarker
AppOutputLines
    AppOutputLine
        AppOutputLines
            ...
        AppOutputLineSymbol
Enumerations
```

For information about creating an instance the `Application` object, see Accessing the API. For reference to the objects exposed by the API, see Object Reference.

17.1.3 Error Handling

The MapForce API returns errors in two different ways. Every API method returns an `HRESULT`. This return value informs the caller about any malfunctions during the execution of the method. If the call was successful, the return value is equal to `S_OK`. C/C++ programmers generally use `HRESULT` to detect errors.

Visual Basic, scripting languages, and other high-level development environments do not give the programmer access to the returning `HRESULT` of a COM call. They use the second error-raising mechanism supported by the MapForce API, the `IErrorInfo` interface. If an error occurs, the API creates a new object that implements the `IErrorInfo` interface. The development

environment takes this interface and fills its own error-handling mechanism with the provided information.

The following text describes how to deal with errors raised from the MapForce API in different development environments.

Visual Basic

A common way to handle errors in Visual Basic is to define an error handler. This error handler can be set with the On Error statement. Usually the handler displays an error message and does some cleanup to avoid spare references and any kind of resource leaks. Visual Basic fills its own Err object with the information from the IErrorInfo interface.

```
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

The Microsoft implementation of JavaScript (JScript) provides a try-catch mechanism to deal with errors raised from COM calls. It is very similar to the VisualBasic approach, in that you also declare an error object containing the necessary information.

```
   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++ gives you easy access to the HRESULT of the COM call and to the IErrorInfo interface.

```
HRESULT hr;

// Call GenerateXSLT() from the MapForce API
if(FAILED(hr = ipDocument->GenerateXSLT()))
{
   IErrorInfo *ipErrorInfo = Null;

   if(SUCCEEDED(::GetErrorInfo(0, &ipErrorInfo)))
   {
      BSTR   bstrDescr;
      ipErrorInfo->GetDescription(&bstrDescr);

      // handle Error information
      wprintf(L"Error message:\t%s\n",bstrDescr);
      ::SysFreeString(bstrDescr);

      // release Error info
      ipErrorInfo->Release();
   }
}
```

17.1.4 Examples

Programming languages differ in the way they support COM access. The following examples for C#, Java, and JScript will help you get started. The code listings in this section are available at the path below:

Windows 10, Windows 8, Windows 7, Windows Vista	C:/Users/<username>/Documents/Altova/MapForce2017/MapForceExamples/API/
Windows XP	C:/Documents and Settings/<username>/My Documents/Altova/MapForce2017/MapForceExamples/API/

17.1.4.1 Example C# Project

After you install MapForce, an example MapForce API client project for C# is available in the following directory:

Windows 10, Windows 8, Windows 7, Windows Vista	C:/Users/<username>/Documents/Altova/MapForce2017/MapForceExamples/API/
Windows XP	C:/Documents and Settings/<username>/My Documents/Altova/MapForce2017/MapForceExamples/API/

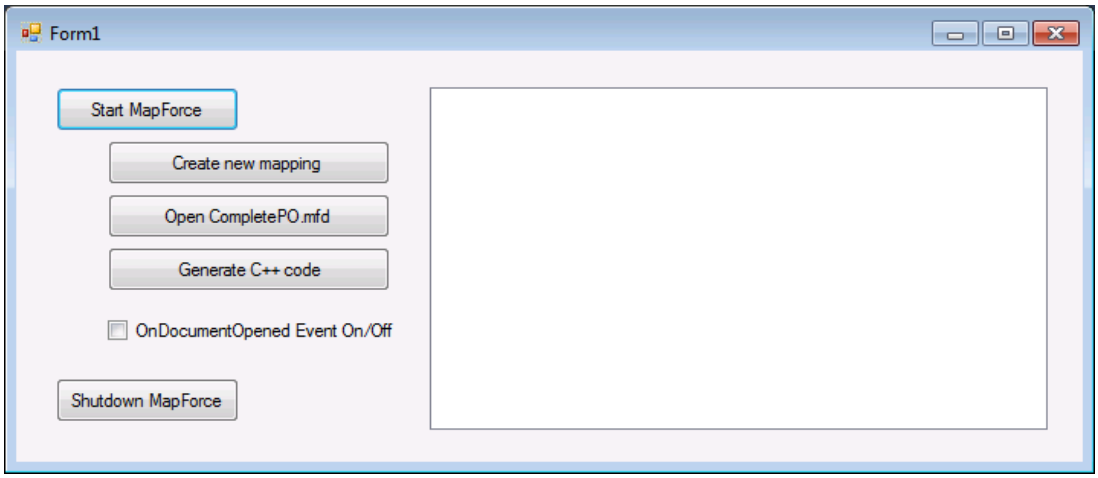
You can compile and run the project with Visual Studio 2008 or later. To compile and run the example, open the solution .sln file in Visual Studio and run **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Note: If you have a 64-bit operating system and are using a 32-bit installation of MapForce, add

the **x86** platform in the solution's Configuration Manager and build the sample using this configuration. A new x86 platform (for the active solution in Visual Studio) can be created in the New Solution Platform dialog (**Build | Configuration Manager | Active solution platform | <New...>**).

When you run the example, a Windows form is displayed, containing buttons that invoke basic MapForce operations:

- Start MapForce
- Create a new mapping design
- Open the CompletePO.mfd file from the **...\MapForceExamples** folder (note that you may need to adjust the path to point to the **\MapForceExamples** folder on your machine)
- Generate C++ code in a temp directory
- Shut down MapForce



Code listing

The listing is commented for ease of understanding. The code essentially consists of a series of handlers for the buttons in the user interface shown above.

```
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 control's 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 ipDocument)
      {
         String sText = "";

         if (ipDocumentName.Length > 0)
            sText = "Document " + ipDocumentName + " was opened!";
         else
            sText = "A new mapping was created.";

         // we need to synchronize the calling thread with the UI thread because
```

```
        // 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);
        }
     }
  }
}
```

17.1.4.2 Example Java Project

After you install MapForce, an example MapForce API client project for Java is available in the following directory:

Windows 10, Windows 8, Windows 7, Windows Vista	`C:/Users/<username>/Documents/Altova/MapForce2017/MapForceExamples/API/`
Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/MapForce2017/MapForceExamples/API/`

You can test the Java example directly from the command line, using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`AltovaAutomation.dll`	Java-COM bridge: DLL part
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`MapForceAPI.jar`	Java classes of the MapForce API
`RunMapForce.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`MapForceAPI_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

What the example does

The example starts up MapForce and performs a few operations, including opening and closing documents. When done, MapForce stays open. You must close it manually.

Running the example from the command line

To run the example from the command line, open a command prompt window, go to the Java folder of the API Examples folder (*see above for location*), and then type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a Java Development Kit (JDK) 7 or later installation on your computer.

Press the **Return** key. The Java source in `RunMapForce.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **File | Import... | General | Existing Projects into Workspace** command to add the Eclipse project file (`.project`) located in the Java folder of the API Examples folder (*see above for location*). The project `RunMapForce` will then appear in your Package Explorer or Navigator.

Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `RunMapForce.java` is listed below with comments.

```
// access general JAVA-COM bridge classes
import java.util.Iterator;

import com.altova.automation.libs.*;

// access XMLSpy Java-COM bridge
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts XMLSpy 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.
         int majorVersionYear = mapforce.getMajorVersion() + 1998;
         String strExamplesFolder = System.getenv("USERPROFILE") + "\
\Documents\\Altova\\MapForce" + Integer.toString(majorVersionYear) + "\
\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");
      }
   }
}
```

17.1.4.3 JScript Examples

After you install MapForce, an example MapForce API client project for JScript is available in the following directory:

Windows 10, Windows 8, Windows 7, Windows Vista	`C:/Users/<username>/Documents/Altova/MapForce2017/MapForceExamples/API/`
Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/MapForce2017/MapForceExamples/API/`

The example files can be run in one of two ways:

- *From the command line:*
  Open a command prompt window and type the name of one of the example scripts (for example, `Start.js`). The Windows Scripting Host that is packaged with Windows XP or later will execute the script.

- *From Windows Explorer:*
  In Windows Explorer, browse for the JScript file and double-click it. The Windows Scripting Host that is packaged with Windows XP or later will execute the script. After the script is executed, the command console gets closed automatically.

The following example files are included:

`Start.js`	Start Mapforce registered as an automation server or connect to a running instance (see Start Application).
`DocumentAccess.js`	Shows how to open, iterate and close documents (see Simple Document Access).
`GenerateCode`	Shows how to invoke code generation using JScript (see Generate Code).

`.js`
`Readme.txt` Provides basic help to run the scripts.

This documentation additionally includes a few extra JScript code listings:

- Example: Code Generation
- Example: Mapping Execution
- Example: Project Support

Start Application

The JScript code listing below starts the application and shuts it down. If an instance of the application is already running, the running instance will be returned.

```
// 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) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// 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
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

Simple Document Access

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and close documents.

```
// 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) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// 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");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") +
"\\My Documents\\Altova\\MapForce" + majorVersionYear + "\\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
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

Generate Code

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and generate C++ 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) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// 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");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") +
"\\My Documents\\Altova\\MapForce" + majorVersionYear + "\\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
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

Example: Code Generation

The following JScript example shows how to load an existing document and generate different kinds of mapping code for it.

```
// ------------------- 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 ---------------------
```

Example: Mapping Execution

The following JScript example shows how to load an existing document with a simple mapping, access its components, set input- and output-instance file names and execute the mapping.

```
/*
   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 ----------------------
```


(C) 2017 Altova GmbH

```
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
{
   // Make MapForce UI visible. This is an API requirement for output
generation.
  mapforce.Visible = true;

   // open an existing mapping.
   // **** adjust the examples path to your needs ! **************
   var sMapForceExamplesPath = fso.BuildPath(
            wshShell.SpecialFolders( "MyDocuments" ),
            "Altova\\MapForce2017\\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 );
}
```

Example: Project Support

The following JScript example shows how to use the MapForce API to automate tasks pertaining to MapForce projects. Before running the example, make sure to edit the variable

`strSamplePath` so that it points to the **MapForceExamples** folder of your MapForce installation.

To successfully run all operations in this example below, you will need the Enterprise version of MapForce. If you have the Professional version running, comment out the lines that insert the WebService project.

```
// ////////////// 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\\MapForce2017\\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_L
anguage +
                                " output folder is " +
objItem.CodeGenSettings_OutputFolder + "\n";
         }
         catch( err ) {}

         // ----- print WSDL settings, if available
         try
         {
            strGlobalText += strTab + "  WSDL File is " + objItem.WSDLFile +
                             " Qualified Name is " + objItem.QualifiedName
+ "\n";
         }
         catch( err ) {}
      }
      catch( ex )
         { strGlobalText += strTab + "[" + objItem.Kind + "]\n" }

      // ---- recurse
      PrintProjectTree( new Enumerator( objItem ), strTab + '   ' );

      objProjectItemIter.moveNext();
   }
}

// ------------------------------------------------------------
// 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;
```

17.2 Object Reference

This section provides reference to the objects of the MapForce COM API. The objects are described in a generic manner, since the API may be used with virtually any language that supports calling a COM object. For language- specific examples, see Examples.

In Java, some syntax variations to the object names exist, as follows:

- ***Classes and class names***
  For every interface of the MapForce automation interface, a Java class exists with the name of the interface.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces, but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with `get` and `set` can be used. If a property does not support write-access, no setter method is available. Example: For the `Name` property of the `Document` interface, the Java methods `getName` and `setName` are available.

- ***Enumerations***
  For every enumeration defined in the automation interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus '`Event`' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `Application`: Java class to access the application
  `ApplicationEvents`: Events interface for the Application
  `ApplicationEventsDefaultHandler`: Default handler for `ApplicationEvents`

17.2.1 Application

The `Application` interface is the interface to a MapForce application object. It represents the main access point for the MapForce application itself. This interface is the starting point to do any further operations with MapForce or to retrieve or create other MapForce related automation objects. For information about creating an instance the `Application` object, see Accessing the API.

Events
Events

Properties and Methods
Properties to navigate the object model:
Application
Parent
Options
Project
Documents

Application status:
Visible

Name
Quit
Status
WindowHandle

MapForce designs:
NewDocument
OpenDocument
OpenURL
ActiveDocument

MapForce projects:
NewProject
OpenProject
ActiveProject

MapForce code generation:
HighlightSerializedMarker

Global resources:
GlobalResourceConfig
GlobalResourceFile

Version information:
Edition
IsAPISupported
MajorVersion
MinorVersion

17.2.1.1 Events

This object supports the following events:

OnDocumentOpened
OnProjectOpened
OnShutdown

OnDocumentOpened

Event: OnDocumentOpened (*iobjDocument* as Document)

Description
This event is triggered when an existing or new document is opened. The corresponding close event is DocumentOnDocumentClosed.

OnProjectOpened

Event: OnProjectOpened (*iobjProject* as Project)

Description
This event is triggered when an existing or new project is loaded into the application. The corresponding close event is ProjectOnProjectClosed.

OnShutdown

Event: OnShutdown ()

Description
This event is triggered when the application is shutting down.

17.2.1.2 ActiveDocument

Property: ActiveDocument as Document (read-only)

Description
Returns the automation object of the currently active document. This property returns the same as Documents.ActiveDocument.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.3 ActiveProject

Property: ActiveProject as Project (read-only)

Description
Returns the automation object of the currently active project.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.4 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.5 Documents

Property: Documents as Documents (read-only)

Description

Returns a collection of all currently open documents.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.6 Edition

Property: Edition as String (read-only)

Description
Returns the edition of the application, e.g. "Altova MapForce Enterprise Edition" for the enterprise edition.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.7 GlobalResourceConfig

Property: GlobalResourceConfig as String

Description
Gets or sets the name of the active global resource configuration file. Per default, the file is called GlobalResources.xml.

The configuration file can be renamed and saved to any location. You can therefore have multiple Global Resources XML files. However, only one of these Global Resources XML File can be active, per application, at one time, and only the definitions contained in this file will be available to the application.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.8 GlobalResourceFile

Property: GlobalResourceFile as String

Description
Gets or sets the global resource definition file. Per default the file is called GlobalResources.xml.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.9 HighlightSerializedMarker

Method: HighlightSerializedMarker(*i_strSerializedMarker* as String)

Description
Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document has not already been loaded, it will be loaded first. See Document.GenerateCodeEx for a method to retrieve a serialized marker.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.
1007 The string passed in *i_strSerializedMarker* is not recognized as a serialized MapForce marker.
1008 The marker points to a location that is no longer valid.

17.2.1.10 IsAPISupported

Property: IsAPISupported as Boolean (read-only)

Description
Returns true if the API is supported in this version of MapForce.

Errors
1001 Invalid address for the return parameter was specified.

17.2.1.11 MajorVersion

Property: MajorVersion as Long (read-only)

Description
Gets the major version number of of MapForce. The version is calculated starting from 1998, and is incremented by one every year. For example, the major version is "18" for the release 2016.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.12 MinorVersion

Property: MinorVersion as Long (read-only)

Description
The minor version number of the product, e.g. 2 for 2006 R2 SP1.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.13 Name

Property: Name as String (read-only)

Description
The name of the application.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.14 NewDocument

Method: NewDocument() as Document

Description
Creates a new empty document. The newly opened document becomes the ActiveDocument.
This method is a shortened form of Documents.NewDocument.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.15 NewProject

Method: NewProject() as Project

Description
Creates a new empty project. The current project is closed. The new project is accessible under ActiveProject.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.16 OpenDocument

Method: OpenDocument(*in strFileName* as String) as Document

Description
Loads a previously saved document file and continues working on it. The newly opened document becomes the ActiveDocument. This method is a shorter form of Documents.OpenDocument.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.17 OpenProject

Method: NewProject() as Project

Description
Opens an existing Mapforce project (*.mfp). The current project is closed. The newly opened project is accessible under ActiveProject.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.18 OpenURL

Method: OpenURL (*istrURL* as String, *istrUser* as String, *istrPassword* as String)

Description
Loads a previously saved document file from an URL location. Allows user name and password to be supplied.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.19 Options

Property: Options as Options (read-only)

Description
This property gives access to options that configure the generation of code.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.20 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
1000 The object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.21 Quit

Method: Quit()

Description
Disconnects from MapForce to allow the application to shutdown. Calling this method is optional since MapForce keeps track of all external COM connections and automatically recognizes a disconnection. For more information on automatic shutdown see the Visible property.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.22 ServicePackVersion

Property: ServicePackVersion as Long (read-only)

Description
The service pack version number of the product, e.g. 1 for 2010 R2 SP1.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.23 Status

Property: Status as Long (read-only)

Description
The status of the application. It is one of the values of the ENUMApplicationStatus enumeration.

Errors
1001 Invalid address for the return parameter was specified.

17.2.1.24 Visible

Property: Visible as Boolean

Description
True if MapForce is displayed on the screen (though it might be covered by other applications or be iconized).

False if MapForce is hidden. The default value for MapForce when automatically started due to a request from the automation server MapForceApplication is false. In all other cases, the property is initialized to true.

An application instance that is visible is said to be controlled by the user (and possibly by clients connected via the automation interface). It will only shut down due to an explicit user request. To

shut down an application instance, set its visibility to false and clear all references to this instance within your program. The application instance will shut down automatically when no further COM clients are holding references to it.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.25 WindowHandle

Property: WindowHandle () as long (read-only)

Description
Retrieve the application's Window Handle.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.2 AppOutputLine

Represents a message line. In contrast to ErrorMarker, its structure is more detailed and can contain a collection of child lines, therefore forming a tree of message lines.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Line access:
GetLineSeverity
GetLineSymbol
GetLineText
GetLineTextEx
GetLineTextWithChildren
GetLineTextWithChildrenEx

A single AppOutputLine consists of one or more sub-lines.
Sub-line access:
GetLineCount

A sub-line consists of one or more cells.
Cell access:
GetCellCountInLine
GetCellIcon
GetCellSymbol
GetCellText
GetCellTextDecoration
GetIsCellText

Below an AppOutputLine there can be zero, one, or more child lines which themselves are of type AppOutputLine, which thus form a tree structure.

Child lines access:
ChildLines

17.2.2.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.2 ChildLines

Property: ChildLines as AppOutputLines (read-only)

Description
Returns a collection of the current line's direct child lines.

Errors
4100 The application object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.3 GetCellCountInLine

Method: GetCellCountInLine (*nLine* as Long) as Long

Description
Gets the number of cells in the sub-line indicated by *nLine* in the current AppOutputLine.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.4 GetCellIcon

Method: GetCellIcon (*nLine* as Long, *nCell* as Long) as Long

Description
Gets the icon of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.5 GetCellSymbol

Method: GetCellSymbol(*nLine* as Long, *nCell* as Long) as AppOutputLineSymbol

Description
Gets the symbol of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.6 GetCellText

Method: GetCellText(*nLine* as Long, *nCell* as Long) as String

Description
Gets the text of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.7 GetCellTextDecoration

Method: GetCellTextDecoration (*nLine* as Long, *nCell* as Long) as Long

Description
Gets the decoration of the text cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.
It can be one of the ENUM AppOutputLine_TextDecoration values.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.8 GetIsCellText

Method: GetIsCellText(*nLine* as Long, *nCell* as Long) as Boolean

Description
Returns true, if the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine* is a text cell.

Errors
4100 The object is no longer valid.

4101 Invalid address for the return parameter was specified.

17.2.2.9 GetLineCount

Method: GetLineCount() as Long

Description
Gets the number of sub-lines the current line consists of.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.10 GetLineSeverity

Method: GetLineSeverity() as Long

Description
Gets the severity of the line. It can be one of the ENUMAppOutputLine_Severity values:

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.11 GetLineSymbol

Method: GetLineSymbol() as AppOutputLineSymbol

Description
Gets the symbol assigned to the whole line.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.12 GetLineText

Method: GetLineText() as String

Description
Gets the contents of the line as text.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.13 GetLineTextEx

Method: GetLineTextEx (*psTextPartSeperator* as String, *psLineSeperator* as String) as String

Description
Gets the contents of the line as text using the specified part and line separators.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.14 GetLineTextWithChildren

Method: GetLineTextWithChildren () as String

Description
Gets the contents of the line including all child and descendant lines as text.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.15 GetLineTextWithChildrenEx

Method: GetLineTextWithChildrenEx (*psPartSep* as String, *psLineSep* as String, *psTabSep* as String, *psItemSep* as String) as String

Description
Gets the contents of the line including all child and descendant lines as text using the specified part, line, tab and item separators.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.16 Parent

Property: Parent as AppOutputLines (read-only)

Description
The parent object according to the object model.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.3 AppOutputLines

Represents a collection of AppOutputLine message lines.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Iterating through the collection:
Count
Item

17.2.3.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.3.2 Count

Property: Count as Integer (read-only)

Description
Retrieves the number of lines in the collection.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.3.3 Item

Property: Item (*nIndex* as Integer) as AppOutputLine (read-only)

Description
Retrieves the line at nIndex from the collection. Indices start with 1.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.3.4 Parent

Property: Parent as AppOutputLine (read-only)

Description
The parent object according to the object model.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.4 AppOutputLineSymbol

An AppOutputLineSymbol represents a link in an AppOutputLine message line which can be clicked in the MapForce Messages window.
It is applied to a cell of an AppOutputLine or to the whole line itself.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Access to AppOutputLineSymbol methods:
GetSymbolHREF
GetSymbolID
IsSymbolHREF

17.2.4.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.2 GetSymbolHREF

Method: GetSymbolHREF () as String

Description
If the symbol is of type URL, returns the URL as a string.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.3 GetSymbolID

Method: GetSymbolHREF () as Long

Description
Gets the ID of the symbol.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.4 IsSymbolHREF

Method: IsSymbolHREF () as Boolean

Description
Indicates if the symbol is of kind URL.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.5 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.5 Component

A Component represents a MapForce component.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Component properties:
HasIncomingConnections
HasOutgoingConnections
CanChangeInputInstanceFile
CanChangeOutputInstanceFile
ComponentName

ID
IsParameterInputRequired
IsParameterSequence
Name
Preview
Schema
SubType
Type

Instance related properties:
InputInstanceFile
OutputInstanceFile

Datapoints:
GetRootDatapoint

Execution:
GenerateOutput

17.2.5.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.2 CanChangeInputInstanceFile

Property: CanChangeInputInstanceFile as Boolean (read-only)

Description
Indicates if the input instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true.
If the component does not have a filename node, false is returned.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.3 CanChangeOutputInstanceFile

Property: CanChangeOutputInstanceFile as Boolean (read-only)

Description
Indicates if the output instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true.
If the component does not have a filename node, false is returned.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.4 ComponentName

Property: ComponentName as String

Description
Gets or sets the component's name.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1246 The component does not support setting its name.
1247 Invalid component name.

17.2.5.5 GenerateOutput

Method: GenerateOutput ([*out*] *pbError* as Boolean) as AppOutputLines

Description
Generates the output file(s) defined in the mapping for the current component only, using a MapForce internal mapping language. The name(s) of the output file(s) are defined as property of the current component which is the output item in the mapping for this generation process.

Remarks
pbError is an output-only parameter. You will receive a value only if the calling language supports output parameters. If not, the value you pass here will remain unchanged when the function has finished.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1248 Generating output is only supported when the graphical user interface is visible.

17.2.5.6 GetRootDatapoint

Method: GetRootDatapoint(side as ENUMComponentDatapointSide, strNamespace as String, strLocalName as String, strParameterName as String) as Datapoint

Description

Gets a root datapoint on the left (input) or right (output) side of a component. To access children and descendants, the Datapoint object provides further methods.

The *side* parameter indicates if an input, or output, datapoint of a component is to be retrieved.

The specified namespace and local name, indicate the specific name of the node whose datapoint is to be retrieved. For components with structural information such as schema components, you will have to provide the namespace together with the local name, or you can just pass an empty string for the namespace.

File-based components like the schema component contain a special node on their root, the filename node. There, GetRootDatapoint can only find the filename node. You will have to pass namespace "`http://www.altova.com/mapforce`" and local name "`FileInstance`" to retrieve a datapoint of this node.

The specified parameter name should be an empty string unless the component in question is a function call component . Since a user-defined function might contain input or output parameters of the same structure, the function call component calling this user-defined function can have more than one root node with an identical namespace and local name.

They will then differ only by their parameter names, which are in fact the names of the according parameter components in the user-defined function mapping itself.

It is not mandatory to specify the parameter name, though. In that case, the method will return the first root datapoint matching the specified namespace and local name.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1231 Datapoint not found.

17.2.5.7 HasIncomingConnections

Property: `HasIncomingConnections` as `Boolean` (read-only)

Description
Indicates if the component has any incoming connections (on its left side) not including the filename node. An incoming connection on the filename node does not have any effect on the returned value.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.8 HasOutgoingConnections

Property: `HasOutgoingConnections` as `Boolean` (read-only)

Description
Indicates if the component has any outgoing connections (on its right side).

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.9 ID

Property: ID as Unsigned Long (read-only)

Description
Retrieves the component ID.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.10 InputInstanceFile

Property: InputInstanceFile as String

Description
Gets or sets the component's input instance file.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.11 IsParameterInputRequired

Property: IsParameterInputRequired as `Boolean`

Description
Gets or sets, if the input parameter component requires an ingoing connection on the function call component of the user-defined function this input parameter component is in.
This property works only for components, which are input parameter components.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1232 This operation works only for an input parameter component.
1240 Changing the document not allowed. It is read-only.

17.2.5.12 IsParameterSequence

Property: IsParameterSequence as `Boolean`

Description
Gets or sets, if the input or output parameter component supports sequences.

This property works only for components, which are input or output parameter components.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1233 This operation works only for an input or output parameter component.
1240 Changing the document not allowed. It is read-only.

17.2.5.13 Name

Property: Name as String (read only)

Description
Gets the component's name.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.14 OutputInstanceFile

Property: OutputInstanceFile as String

Description
Gets or sets the component's output instance file.

Trying to access the OutputInstanceFile of a component via the API does not return any data if the "File" connector of the component has been connected to another item in the mapping.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.15 Parent

Property: Parent as Mapping (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.16 Preview

Property: Preview as `Boolean`

Description
Gets or sets, if the component is the current preview component.

This property works only for components, which are target components in the document's main mapping. Only one target component in the main mapping can be the preview component at any time.

When setting this property, it is only possible to set it to true. This then will also implicitly set the Preview property of all other components to false.

If there is just a single target component in the main mapping, it is also the preview component.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1234 Only a target component in the main mapping can be set as preview component.
1235 A component cannot be set as non-preview component. Set another component as preview component instead.

17.2.5.17 Schema

Property: Schema as String (read-only)

Description
Retrieves the component's schema file name.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.18 SubType

Property: SubType as ENUMComponentSubType (read-only)

Description
Retrieves the component's sub type.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.19 Type

Property: Type as ENUM ComponentType (read-only)

Description
Retrieves the component's type.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.20 UsageKind

Property: UsageKind as ENUM UsageKind (read-only)

Description
Retrieves the component's usage kind.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6 Components

Represents a collection of Component objects.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Iterating through the collection:
Count
Item

17.2.6.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.2 Count

Property: Count as Integer (read-only)

Description
Retrieves the number of components in the collection.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.3 Item

Property: Item (*nIndex* as Integer) as Component (read-only)

Description
Retrieves the component at nIndex from the collection. Indices start with 1.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.4 Parent

Property: Parent as Mapping (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.7 Connection

A Connection object represents a connector between two components.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Properties
ConnectionType

17.2.7.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
2100 The object is no longer valid.
2101 Invalid address for the return parameter was specified.

17.2.7.2 ConnectionType

Property: ConnectionType as ENUMConnectionType

Description
Gets or sets the connection's type.

Errors
2100 The application object is no longer valid.
2101 Invalid address for the return parameter was specified.
2102 Changing the document not allowed. It is read-only.
2103 Failed changing connection type.

17.2.7.3 Parent

Property: Parent as Mapping (read-only)

Description
The parent object according to the object model.

Errors
2100 The object is no longer valid.
2101 Invalid address for the return parameter was specified.

17.2.8 Datapoint

A Datapoint object represents an input or output icon of a component.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Methods
GetChild

17.2.8.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
2000 The object is no longer valid.
2001 Invalid address for the return parameter was specified.

17.2.8.2 GetChild

Method: GetChild(strNamespace as String, strLocalName as String, searchFlags as ENUMSearchDatapointFlags) as Datapoint

Description
Scans for a direct child datapoint of the current datapoint, by namespace and local name.

Search flags can be passed as combination of values (combined using binary OR) of the ENUMSearchDatapointFlags enumeration.

A schema component with elements that contain mixed content, each display an additional child node, the so-called text() node. To retrieve a datapoint of a text() node, you will have to pass an empty string in strNamespace as well as `"#text"` in strLocalName and eSearchDatapointElement in searchFlags.

Errors
2000 The application object is no longer valid.
2001 Invalid address for the return parameter was specified.
2002 Datapoint not found.

17.2.8.3 Parent

Property: Parent as Component (read-only)

Description
The parent object according to the object model.

Errors
2000 The object is no longer valid.
2001 Invalid address for the return parameter was specified.

17.2.9 Document

A Document object represents a MapForce document (a loaded MFD file).
A document contains a main mapping and zero or more local user-defined-function mappings.

Events
Events

Properties and Methods
Properties to navigate the object model:
Application
Parent

File handling:
Activate
Close
FullName
Name
Path
Saved
Save
SaveAs

Mapping handling:
MainMapping
Mappings
CreateUserDefinedFunction

Component handling:
FindComponentByID

Code generation:
OutputSettings_ApplicationName
JavaSettings_BasePackageName

GenerateCHashCode
GenerateCodeEx
GenerateCppCode
GenerateJavaCode
GenerateXQuery
GenerateXSLT
GenerateXSLT2
HighlightSerializedMarker

Mapping execution:
GenerateOutput
GenerateOutputEx

View access:
MapForceView

Obsolete:
OutputSettings_Encoding

17.2.9.1 Events

This object supports the following events:

OnDocumentClosed
OnModifiedFlagChanged

OnDocumentClosed

Event: OnDocumentClosed (*iobjDocument* as Document)

Description
This event is triggered when a document is closed. The document object passed into the event handler should not be accessed. The corresponding open event is Application.OnDocumentOpened.

OnModifiedFlagChanged

Event: OnModifiedFlagChanged (*ibIsModified* as Boolean)

Description
This event is triggered when a document's modification status changes.

17.2.9.2 Activate

Method: Activate ()

Description
Makes this document the active document.

Errors
1200 The application object is no longer valid.

17.2.9.3 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.4 Close

Method: Close ()

Description
Closes the document without saving.

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.5 CreateUserDefinedFunction

Method: CreateUserDefinedFunction(strFunctionName as String, strLibraryName as String, strSyntax as String, strDetails as String, bInlinedUse as Boolean) as Mapping

Description
Creates a user defined function in the current document.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1208 Failed creating user-defined function.
1209 Changing the document not allowed. It is read-only.

17.2.9.6 FindComponentByID

Method: FindComponentByID (*nID* as Unsigned Long) as Component

Description
Searches in the whole document, so all its mappings, for the component with the specified id.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.7 FullName

Property: FullName as String

Description
Path and name of the document file.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.8 GenerateCHashCode

Method: GenerateCHashCode ()

Description
Generate C# code that will perform the mapping. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.9 GenerateCodeEx

Method: GenerateCodeEx (*inLanguage* as ENUMProgrammingLanguage) as ErrorMarkers

Description
Generates code that will perform the mapping. The parameter *inLanguage* specifies the target language. The method returns an object that can be used to enumerate all messages created by the code generator. These are the same messages that get displayed in the Messages window of MapForce.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.10 GenerateCppCode

Method: GenerateCppCode ()

Description
Generates C++ code that will perform the mapping. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.11 GenerateJavaCode

Method: GenerateJavaCode ()

Description
Generates Java code that will perform the mapping. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.12 GenerateOutput

Method: GenerateOutput ()

Description
Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1206 Error during execution of mapping algorithm.
1210 Generating output is only supported when the graphical user interface is visible.

This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

17.2.9.13 GenerateOutputEx

Method: GenerateOutputEx () as AppOutputLines

Description
Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping. This method is identical to GenerateOutput except for its return value containing the resulting messages, warnings and errors arranged as trees of AppOutputLines.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1206 Error during execution of mapping algorithm.
1210 Generating output is only supported when the graphical user interface is visible.

This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

17.2.9.14 GenerateXQuery

Method: GenerateXQuery ()

Description
Generates mapping code as XQuery. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/ XSLT2/ XQuery code generation.

17.2.9.15 GenerateXSLT

Method: GenerateXSLT ()

Description
Generates mapping code as XSLT. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/ XSLT2/ XQuery code generation.

17.2.9.16 GenerateXSLT2

Method: GenerateXSLT2 ()

Description
Generates mapping code as XSLT2. Uses the properties defined in ApplicationOptions to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/ XSLT2/ XQuery code generation.

17.2.9.17 HighlightSerializedMarker

Method: HighlightSerializedMarker (*i_strSerializedMarker* as String)

Description
Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document is not already loaded, it will be loaded first. See GenerateCodeEx for a method to retrieve a serialized marker.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.
1007 The string passed in *i_strSerializedMarker* is not recognized a serialized MapForce marker.
1008 The marker points to a location that is no longer valid.

17.2.9.18 JavaSettings_BasePackageName

Property: JavaSettings_BasePackageName as String

Description
Sets or retrieves the base package name used when generating Java code. This property is available in UI- dialog for the Document Settings.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.19 MainMapping

Property: MainMapping as Mapping (read-only)

Description
Retrieves the main mapping of the document.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.20 MapForceView

Property: MapForceView as MapForceView (read-only)

Description
This property gives access to functionality specific to the MapForce view.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.21 Mappings

Property: Mappings as Mappings (read-only)

Description
Returns a collection of the mappings contained in the document.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.22 Name

Property: Name as String

Description
Name of the document file without file path.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.23 OutputSettings_ApplicationName

Property: OutputSettings_ApplicationName as String

Description
Sets or retrieves the application name available in the Document Settings dialog.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.24 OutputSettings_Encoding (obsolete)

Property: OutputSettings_Encoding as String

Description
obsolete
This property is not supported anymore. Mapping output encoding settings do not exist anymore. Components have individual output encoding settings.

17.2.9.25 Parent

Property: Parent as Documents (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.26 Path

Property: Path as String

Description
Path of the document file without name.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.27 Save

Method: Save ()

Description
Save the document to the file defined by DocumentFullName.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.28 SaveAs

Method: SaveAs (*lstrFileName* as String)

Description
Save document to specified file name, and set DocumentFullName to this value if save operation was successful.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.29 Saved

Property: Saved as Boolean (read-only)

Description
True if the document was not modified since the last save operation, false otherwise.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.10 Documents

Represents a collection of Document objects.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Open and create mappings:
OpenDocument
NewDocument

Iterating through the collection:
Count
Item
ActiveDocument

17.2.10.1 ActiveDocument

Property: ActiveDocument as Document (read-only)

Description
Retrieves the active document. If no document is open, null is returned.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.2 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.3 Count

Property: Count as Integer (read-only)

Description
Retrieves the number of documents in the collection.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.4 Item

Property: Item (*nIndex* as Integer) as Document (read-only)

Description
Retrieves the document at nIndex from the collection. Indices start with 1.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.5 NewDocument

Method: NewDocument() as Document

Description
Creates a new document, adds it to the end of the collection, and makes it the active document.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.6 OpenDocument

Method: OpenDocument(*strFilePath* as String) as Document

Description
Opens an existing mapping document (*.mfd). Adds the newly opened document to the end of the collection and makes it the active document.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.7 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.11 ErrorMarker

Represents a simple message line. In difference to AppOutputLine, error markers do not have a hierarchical structure.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Access to message information:
DocumentFileName
ErrorLevel
Highlight
Serialization

Text

17.2.11.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.2 DocumentFileName

Property: DocumentFileName as String (read-only)

Description
Retrieves the name of the mapping file that the error marker is associated with.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.3 ErrorLevel

Property: ErrorLevel as ENUMCodeGenErrorLevel (read-only)

Description
Retrieves the severity of the error.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.4 Highlight

Method: Highlight()

Description
Highlights the item that the error marker is associated with. If the corresponding document is not open, it will be opened.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.
1008 The marker points to a location that is no longer valid.

17.2.11.5 Serialization

Property: Serialization as String (read-only)

Description
Serialize error marker into a string. Use this string in calls to Application.HighlightSerializedMarker or Document.HighlightSerializedMarker to highlight the marked item in the mapping. The string can be persisted and used in other instantiations of MapForce or its Control.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.6 Text

Property: Text as String (read-only)

Description
Retrieves the message text.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.7 Parent

Property: Parent as ErrorMarkers (read-only)

Description
The parent object according to the object model.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.12 ErrorMarkers

Represents a collection of ErrorMarker objects.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Iterating through the collection:
Count
Item

17.2.12.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.12.2 Count

Property: Count as Integer (read-only)

Description
Retrieves the number of error markers in the collection.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.12.3 Item

Property: Item (*nIndex* as Integer) as ErrorMarker (read-only)

Description
Retrieves the error marker at nIndex from the collection. Indices start with 1.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.12.4 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.13 MapForceView

Represents the current view in the MapForce Mapping tab for a document.
A document has exactly one MapForceView which displays the currently active mapping.

Properties and Methods
Properties to navigate the object model:
Application
Parent

View activation and view properties:
Active
ShowItemTypes
ShowLibraryInFunctionHeader
HighlightMyConnections
HighlightMyConnectionsRecursively

Mapping related properties:
ActiveMapping
ActiveMappingName

Adding items:
InsertWSDLCall
InsertXMLFile
InsertXMLSchema
InsertXMLSchemaWithSample

17.2.13.1 Active

Property: Active as Boolean

Description
Use this property to query if the mapping view is the active view, or set this view to be the active one.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.2 ActiveMapping

Property: ActiveMapping as Mapping

Description
Gets or sets the currently active mapping in the document this MapForceView belongs to.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.3 ActiveMappingName

Property: ActiveMappingName as String

Description
Gets or sets the currently active mapping by name in the document this MapForceView belongs to.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.4 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.5 HighlightMyConnections

Property: HighlightMyConnections as Boolean

Description
This property defines whether connections from the selected item only should be highlighed.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.6 HighlightMyConnectionsRecursivey

Property: HighlightMyConnectionsRecursively as Boolean

Description
This property defines if only the connections coming directly or indirectly from the selected item should be highlighed.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.7 InsertWSDLCall

Method: InsertWSDLCall(*istrWSDLFileName* as String)

Description
Adds a new WSDL call component to the mapping.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.8 InsertXMLFile (obsolete)

Method: InsertXMLFile (*istrXMLFileName* as String,*istrRootElement* as String)

Description
obsolete
MapForceView.InsertXMLFile is obsolete. Use Mapping.InsertXMLFile instead.

Adds a new component to the mapping.

The component's internal structure is determined by the schema referenced in the specified XML file.

The second parameter defines the root element of this schema, if there is more than one candidate.
When passing an empty string as root element, the root element of the xml file will be used. Otherwise if more candidates are available, a Select Root Element dialog will pop up regardless if MapForce is visible or not.

The specified XML file is used as the input sample to evaluate the mapping.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.9 InsertXMLSchema (obsolete)

Method: InsertXMLSchema (*istrSchemaFileName* as String,*istrRootElement* as String)

Description
obsolete
MapForceView.InsertXMLSchema is obsolete. Use Mapping.InsertXMLSchema instead.

Adds a new component to the mapping.

The component's internal structure is determined by the specified schema file.

The second parameter defines the root element of this schema if there is more than one candidate.
If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up regardless if MapForce is visible or not.

No XML input sample is assigned to this component.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.10 InsertXMLSchemaWithSample (obsolete)

Method: InsertXMLSchemaWithSample (*istrSchemaFileName* as String, *istrXMLSampleName* as String, *istrRootElement* as String)

Description
obsolete
MapForceView.InsertXMLSchemaWithSample is obsolete. Use Mapping.InsertXMLFile instead. Notice, Mapping.InsertXMLFile does not require a parameter for passing the root element. The root element is automatically set as the xml file's root element name.

Adds a new component to the mapping.

The component's internal structure is determined by the specified schema file.

The second parameter is stored as the XML input sample for mapping evaluation.

The third parameter defines the root element of this schema if there is more than one candidate. When passing an empty string as root element, the root element of the xml sample file will be used.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.11 Parent

Property: Parent as Document (read-only)

Description
The parent object according to the object model.

Errors
1300 The object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.12 ShowItemTypes

Property: ShowItemTypes as Boolean

Description
This property defines if types of items should be shown in the mapping diagram.

Errors
1300 The application object is no longer valid.

1301 Invalid address for the return parameter was specified.

17.2.13.13 ShowLibraryInFunctionHeader

Property: ShowLibraryInFunctionHeader as Boolean

Description
This property defines whether the name of the function library should be part of function names.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.14 Mapping

A Mapping object represents a mapping in a document, so the main mapping, or a local user-defined-function mapping.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Mapping properties:
IsMainMapping
Name

Components in the mapping:
Components

Adding items:
CreateConnection
InsertFunctionCall
InsertXMLFile
InsertXMLSchema
InsertXMLSchemaInputParameter
InsertXMLSchemaOutputParameter

17.2.14.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.2 Components

Property: Components as Components (read-only)

Description
Returns a collection of all components in the current mapping.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.3 CreateConnection

Method: CreateConnection(DatapointFrom as Datapoint, DatapointTo as Datapoint) as Connection

Description
Creates a connection between the two supplied datapoints (DatapointFrom & DatapointTo).

It will fail to do so if the DatapointFrom is not an output-side datapoint, the DatapointTo is not an input-side datapoint, or a connection between these two datapoints already exists.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1241 Failed creating the connection.

17.2.14.4 InsertFunctionCall

Method: InsertFunctionCall(strFunctionName as String, strLibraryName as String) as Component

Description
Inserts a function call component into the current mapping.

The specified library and function names indicate the function or user-defined function to be called.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1242 Failed creating function call component.

17.2.14.5 InsertXMLFile

Method: InsertXMLFile(i_strFileName as String, i_strSchemaFileName as String) as Component

Description
Adds a new XML schema component to the mapping.

The component's internal structure is determined by the schema referenced in the specified XML file (i_strFileName) or, if the XML file does not reference a schema file, by the separately specified schema file (i_strSchemaFileName).

If the XML file has a schema file reference, then the parameter i_strSchemaFileName is ignored.

The root element of the XML file will be used in the component.

The specified XML file is used as the input sample to evaluate the mapping.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1244 Failed creating component.

17.2.14.6 InsertXMLSchema

Method: InsertXMLSchema(i_strSchemaFileName as String, i_strXMLRootName as String) as Component

Description
Adds a new XML schema component to the mapping.

The component's internal structure is determined the specified schema file.

The second parameter defines the root element of this schema if there is more than one candidate.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

No XML input sample is assigned to this component.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1244 Failed creating component.

17.2.14.7 InsertXMLSchemaInputParameter

Method: InsertXMLSchemaInputParameter(strParamName as String, strSchemaFileName as String, strXMLRootElementName as String) as Component

Description
Inserts an XML schema input parameter component into the current mapping.

The current mapping has to be a **user- defined** function. Trying to insert it (the schema input parameter) into the **main mapping** will fail.

strParamName is the name of the input parameter component to create and **strSchemaFileName** and **strXMLRootElementName** indicate the respective schema file and the root element of the schema file to be used.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1243 Failed creating parameter component.
1245 This operation is not supported for the main mapping.

17.2.14.8 InsertXMLSchemaOutputParameter

Method: InsertXMLSchemaOutputParameter(strParamName as String, strSchemaFileName as String, strXMLRootElementName as String) as Component

Description
Inserts an XML schema output parameter component into the current mapping.

The current mapping has to be a **user- defined** function. Trying to insert it (the schema output paramter) into the **main** mapping will fail.

strParamName is the name of the output parameter component to create and **strSchemaFileName** and **strXMLRootElementName** indicate the schema file and the root element of the schema file to be used respectively.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1243 Failed creating parameter component.
1245 This operation is not supported for the main mapping.

17.2.14.9 IsMainMapping

Property: IsMainMapping as Boolean (read- only)

Description
Indicates if the current mapping is the main mapping of the document the mapping is in.

True means it is the main mapping.

False means it is a user defined function (UDF).

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.10 Name

Property: Name as String (read-only)

Description
The name of the mapping / user defined function (UDF).

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.11 Parent

Property: Parent as Document (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15 Mappings

Represents a collection of Mapping objects.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Iterating through the collection:
Count
Item

17.2.15.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15.2 Count

Property: Count as Integer (read-only)

Description
Retrieves the number of mappings in the collection.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15.3 Item

Property: Item (*nIndex* as Integer) as Mapping (read-only)

Description
Retrieves the mapping at nIndex from the collection. Indices start with 1.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15.4 Parent

Property: Parent as Document (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.16 Options

This object gives access to all MapForce options available in the **Tools | Options** dialog.

Properties and Methods
Properties to navigate the object model:
Application
Parent

General options:
ShowLogoOnPrint

ShowLogoOnStartup
UseGradientBackground

Options for code generation:
DefaultOutputEncoding
DefaultOutputByteOrder
DefaultOutputByteOrderMark
XSLTDefaultOutputDirectory
CodeDefaultOutputDirectory
CPPSettings_DOMType
CPPSettings_GenerateVC6ProjectFile
CppSettings_GenerateVSProjectFile
CPPSettings_LibraryType
CPPSettings_UseMFC
CSharpSettings_ProjectType

17.2.16.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.2 CodeDefaultOutputDirectory

Property: CodeDefaultOutputDirectory as String

Description
Specifies the target directory where files generated by DocumentGenerateCppCode, DocumentGenerateJavaCode and DocumentGenerateCHashCode, are placed.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.3 CPPSettings_DOMType

Property: CPPSettings_DOMType as ENUMDOMType

Description
Specifies the DOM type used by DocumentGenerateCppCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.
1402 The parameter value is out of range
1403 The parameter value is not available anymore

17.2.16.4 CPPSettings_GenerateVC6ProjectFile

Property: CPPSettings_GenerateVC6ProjectFile as Boolean

Description
Specifies if VisualC++ 6.0 project files should be generated by DocumentGenerateCppCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.5 CppSettings_GenerateVSProjectFile

Property: CppSettings_GenerateVSProjectFile as ENUMProjectType

Description
Specifies which version of VisualStudio (2005 / 2008 / 2010) project files should be generated by DocumentGenerateCppCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.
1402 The parameter value is out of range
1403 The paramater value is not available anymore

17.2.16.6 CPPSettings_LibraryType

Property: CPPSettings_LibraryType as ENUMLibType

Description
Specifies the library type used by DocumentGenerateCppCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.7 CPPSettings_UseMFC

Property: CPPSettings_UseMFC as Boolean

Description
Specifies if MFC support should be used by C++ code generated by DocumentGenerateCppCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.8 CSharpSettings_ProjectType

Property: CSharpSettings_ProjectType as ENUMProjectType

Description
Specifies the type of C# project used by DocumentGenerateCHashCode.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.
1402 The parameter value is out of range
1403 The paramater value is not available anymore

17.2.16.9 DefaultOutputByteOrder

Property: DefaultOutputByteOrder as String

Description
Byte order for the file encoding used for output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.10 DefaultOutputByteOrderMark

Property: DefaultOutputByteOrderMark as Boolean

Description
Indicates if a byte order mark (BOM), is to be included in the file encoding of output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.11 DefaultOutputEncoding

Property: DefaultOutputEncoding as String

Description
File encoding used for output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.12 GenerateWrapperClasses

Property: GenerateWrapperClasses as Boolean

Description
Indicates if wrapper classes are also to be generated when generating code.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.13 JavaSettings_ApacheAxisVersion (obsolete)

Property: JavaSettings_ApacheAxisVersion as ENUMApacheAxisVersion

Description
Specifies the Apache Axis version to use when generating Java code for web service implementations with SOAP 1.1.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.14 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
1400 The object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.15 ShowLogoOnPrint

Property: ShowLogoOnPrint as Boolean

Description
Show or hide the MapForce logo on printed outputs.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.16 ShowLogoOnStartup

Property: ShowLogoOnStartup as Boolean

Description
Show or hide the MapForce logo on application startup.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.17 UseGradientBackground

Property: UseGradientBackground as Boolean

Description
Set or retrieve the background color mode for a mapping window.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.18 XSLTDefaultOutputDirectory

Property: XSLTDefaultOutputDirectory as String

Description
Specifies the target directory where files generated by DocumentGenerateXSLT are placed.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.17 Project

A Project object represents a project and its tree of project items in MapForce.

Events
Events

Properties and Methods
Properties to navigate the object model:
Application
Parent

File handling:
FullName
Name

Path
Saved
Save
Close

Project tree navigation:
Count
Item
_NewEnum

Project tree manipulation:
AddActiveFile
AddFile
InsertWebService (Enterprise edition only)
CreateFolder

Code-generation:
Output_Folder
Output_Language
Output_TextEncoding
Java_BasePackageName
GenerateCode
GenerateCodeEx
GenerateCodeIn
GenerateCodeInEx

For examples of how to use the properties and methods listed above, see Example: Project Support. For operations with Web services, the MapForce Enterprise edition is required.

17.2.17.1 Events

This object supports the following events:

OnProjectClosed

OnProjectClosed

Event: OnProjectClosed (*iobProject* as Project)

Description
This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is ApplicationOnProjectOpened.

17.2.17.2 _NewEnum

Property: _NewEnum () as IUnknown (read-only)

Description
This property supports language-specific standard enumeration.

Errors

1500 The object is no longer valid.

Examples

```
// ------------------------------------------------------------
// JScript sample - enumeration of a project's project items.
function AllChildrenOfProjectRoot()
{
   objProject = objMapForceActiveProject;
   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 = objMapForceActiveProject;
   if (objProject != null)
   {
      IterateProjectItemsRec(new Enumerator(objProject));
   }
}
```

17.2.17.3 AddActiveFile

Method: AddActiveFile () as ProjectItem

Description
Adds the currently open document to the mapping folder of the project's root.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.
1503 No active document is available.
1504 Active documents needs to be given a path name before it can be added to the project.
1705 Mapping could not be assigned to project. Maybe it is already contained in the target folder.

17.2.17.4 AddFile

Method: AddFile (*istrFileName* as String) as ProjectItem

Description
Adds the specified document to the mapping folder of the project's root.

Errors
1500 The object is no longer valid.
1501 The file name is empty.
Invalid address for the return parameter was specified.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.17.5 Application

Property: Application as Application (read-only)

Description
Retrieves the top-level application object.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.6 Close

Method: Close ()

Description
Closes the project without saving.

Errors
1500 The object is no longer valid.

17.2.17.7 Count

Property: Count as Integer (read-only)

Description
Retrieves number of children of the project's root item.

Errors
1500 The object is no longer valid.

Examples
See Item or _NewEnum.

17.2.17.8 CreateFolder

Method: CreateFolder(*istrFolderName* as String)as ProjectItem

Description
Creates a new folder as a child of the project's root item.

Errors
1500 The object is no longer valid.
1501 Invalid folder name or invalid address for the return parameter was specified.

17.2.17.9 FullName

Property: FullName as String (read-only)

Description
Path and name of the project file.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.10 GenerateCode

Method: GenerateCode ()

Description
Generates code for all project items of the project. The code language and output location is determined by properties of the project and project items.

Errors
1500 The object is no longer valid.
1706 Error during code generation

17.2.17.11 GenerateCodeEx

Method: GenerateCode () as ErrorMarkers

Description
Generates code for all project items of the project. The code language and output location are 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.

Errors
1500 The object is no longer valid.

(C) 2017 Altova GmbH

1501 Invalid address for the return parameter was specified.
1706 Error during code generation

17.2.17.12 GenerateCodeIn

Method: GenerateCodeIn (*inLanguage* as ENUMProgrammingLanguage)

Description
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.

Errors
1500 The object is no longer valid.
1706 Error during code generation

17.2.17.13 GenerateCodeInEx

Method: GenerateCodeIn (*inLanguage* as ENUMProgrammingLanguage) as ErrorMarkers

Description
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.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.
1706 Error during code generation

17.2.17.14 InsertWebService

Method: InsertWebService (*istrWSDLFile* as String, *istrService* as String, *istrPort* as String, *ibGenerateMappings* as Boolean) as ProjectItem

Description
Inserts a new Web service project into the project's Web service folder. If ibGenerateMappings is true, initial mapping documents for all ports get generated automatically.

Errors
1500 The object is no longer valid.
1501 WSDL file can not be found or is invalid.
Service or port names are invalid.
Invalid address for the return parameter was specified.
1503 Operation not supported by current edition.

17.2.17.15 Item

Property: Item(*inItemIndex* as Integer) as ProjectItem (read-only)

Description
Returns the child at *inItemIndex* position of the project's root. The index is zero-based. The largest valid index is Count-1. For an alternative to visit all children see _NewEnum.

Errors
1500 The object is no longer valid.

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
}
```

17.2.17.16 Java_BasePackageName

Property: Java_BasePackageName as String

Description
Sets or gets the base package name of the Java packages that will be generated. This property is used only when generating Java code.

Errors
1500 The object is no longer valid.
1501 Invalid package name specified.
Invalid address for the return parameter was specified.

17.2.17.17 Name

Property: Name as String (read-only)

Description
Name of the project file without file path.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.18 Output_Folder

Property: Output_Folder as String

Description
Sets or gets the default output folder used with GenerateCode and GenerateCodeIn. Project items can overwrite this value in their CodeGenSettings_OutputFolder property, when CodeGenSettings_UseDefault is set to false.

Errors
1500 The object is no longer valid.
1501 Invalid folder name specified.
Invalid address for the return parameter was specified.

17.2.17.19 Output_Language

Property: Output_Language as ENUMProgrammingLanguage

Description
Sets or gets the default language for code generation when using GenerateCode. Project items can overwrite this value in their CodeGenSettings_OutputLanguage property, when CodeGenSettings_UseDefault is set to false.

Errors
1500 The object is no longer valid.
1501 Invalid language specified.
Invalid address for the return parameter was specified.

17.2.17.20 Output_TextEncoding

Property: Output_TextEncoding as String

Description
Sets or gets the text encoding used when generating XML-based code.

Errors
1500 The object is no longer valid.
1501 Invalid text encoding specified.
Invalid address for the return parameter was specified.

17.2.17.21 Parent

Property: Parent as Application (read-only)

Description
The parent object according to the object model.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.22 Path

Property: Path as String (read-only)

Description
Path of the project file without name.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.23 Save

Method: Save ()

Description
Saves the project to the file defined by FullName.

Errors
1500 The object is no longer valid.
1502 Can't save to file.

17.2.17.24 Saved

Property: Saved as Boolean (read-only)

Description
True if the project was not modified since the last Save operation, false otherwise.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.18 ProjectItem

A ProjectItem object represents one item in a project tree.

Properties and Methods
Properties to navigate the object model:
Application
Parent

Project tree navigation:
Count
Item
_NewEnum

Project item properties:

Kind
Name
WSDLFile (only available to Web service project items)
QualifiedName (only available to Web service project items)

Project tree manipulation:
AddActiveFile (only available to folder items)
AddFile (only available to folder items)
CreateFolder (only available to folder items)
CreateMappingForProject (only available to Web service operations)
Remove

Document access:
Open (only available to mapping items and Web service operations)

Code-generation:
CodeGenSettings_UseDefault
CodeGenSettings_OutputFolder
CodeGenSettings_Language
GenerateCode
GenerateCodeEx
GenerateCodeIn
GenerateCodeInEx

For examples of how to use the properties and methods listed above, see Example: Project Support. For operations with Web services, the MapForce Enterprise edition is required.

17.2.18.1 _NewEnum

Property: _NewEnum () as IUnknown (read-only)

Description
This property supports language specific standard enumeration.

Errors
1700 The object is no longer valid.

Examples
See ProjectItem or Project_NewEnum.

17.2.18.2 AddActiveFile

Method: AddActiveFile () as ProjectItem

Description
Adds the currently active document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Errors
1700 The object is no longer valid.
1701 The file name is empty.
Invalid address for the return parameter was specified.
1703 No active document is available.
1704 Active documents needs to be given a path name before it

can be added to the project.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.18.3 AddFile

Method: AddFile (*istrFileName* as String) as ProjectItem

Description
Adds the specified document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Errors
1700 The object is no longer valid.
1701 The file name is empty.
Invalid address for the return parameter was specified.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.18.4 Application

Property: Application as Application (read-only)

Description
Retrieves the top-level application object.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.5 CodeGenSettings_Language

Property: CodeGenSettings_Language as ENUMProgrammingLanguage

Description
Gets or sets the language to be used with GenerateCode or ProjectGenerateCode. This property is consulted only if CodeGenSettings_UseDefault is set to false.

Errors
1700 The object is no longer valid.
1701 Invalid language or invalid address for the return parameter was specified.

17.2.18.6 CodeGenSettings_OutputFolder

Property: CodeGenSettings_OutputFolder as String

Description
Gets or sets the output directory to be used with GenerateCode, GenerateCodeIn, Project.GenerateCode or Project.GenerateCodeIn. This property is consulted only if CodeGenSettings_UseDefault is set to false.

Errors
1700 The object is no longer valid.
1701 An invalid output folder or an invalid address for the return parameter was specified.

17.2.18.7 CodeGenSettings_UseDefault

Property: CodeGenSettings_UseDefault as Boolean

Description
Gets or sets whether output directory and code language are used as defined by either (a) the parent folders, or (b) the project root. This property is used with calls to GenerateCode, GenerateCodeIn, Project.GenerateCode and Project.GenerateCodeIn. If this property is set to false, the values of CodeGenSettings_OutputFolder and CodeGenSettings_Language are used to generate code for this project item..

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.8 Count

Property: Count as Integer (read-only)

Description
Retrieves number of children of this project item. Also see Item.

Errors
1700 The object is no longer valid.

Examples
See Project.Item or Project.NewEnum.

17.2.18.9 CreateFolder

Method: CreateFolder(*istrFolderName* as String) as ProjectItem

Description
Creates a new folder as a child of this project item.

Errors
1700 The object is no longer valid.
1701 Invalid folder name or invalid address for the return parameter was specified.
1702 The project item does not support children.

17.2.18.10 CreateMappingForProject

Method: CreateMappingForProject(*istrFileName* as String) as ProjectItem

Description
Creates an initial mapping document for a Web service operation and saves it to istrFileName. When using ProjectInsertWebService you can use the ibGenerateMappings flag to let MapForce automatically generate initial mappings for all ports.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1707 Cannot create new mapping.
The project item does not support auto-creation of initial mappings or a mapping already exists.
1708 Operation not supported in current edition.

17.2.18.11 GenerateCode

Method: GenerateCode ()

Description
Generates code for this project item and its children. The code language and output location is determined by CodeGenSettings_UseDefault, CodeGenSettings_Language and CodeGenSettings_OutputFolder. Children of this project item can have their own property settings related to code-generation.

Errors
1700 The object is no longer valid.
1706 Error during code generation.

17.2.18.12 GenerateCodeEx

Method: GenerateCode () as ErrorMarkers

Description
Generates code for this project item and its children. The code language and output location are determined by CodeGenSettings_UseDefault, CodeGenSettings_Language and CodeGenSettings_OutputFolder. Children of this project item can have their own property settings related to code-generation.

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.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1706 Error during code generation.

17.2.18.13 GenerateCodeIn

Method: GenerateCodeIn (*inLanguage* as ENUMProgrammingLanguage)

Description
Generates code for the project item and its children in the specified language. The output location is determined by CodeGenSettings_UseDefault and CodeGenSettings_OutputFolder. Children of this project item can have their own property settings related to code-generation.

Errors
1700 The object is no longer valid.
1701 Invalid language specified.
1706 Error during code generation.

17.2.18.14 GenerateCodeInEx

Method: GenerateCodeIn (*inLanguage* as ENUMProgrammingLanguage) as ErrorMarkers

Description
Generates code for the project item and its children in the specified language. The output location is determined by CodeGenSettings_UseDefault and CodeGenSettings_OutputFolder. Children of this project item can have their own property settings related to code-generation.

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.

Errors
1700 The object is no longer valid.
1701 Invalid language specified or invalid address for the return parameter was specified.
1706 Error during code generation.

17.2.18.15 Item

Property: Item (*inItemIndex* as Integer) as ProjectItem (read-only)

Description
Returns the child at inItemIndex position of this project item. The index is zero-based. The largest valid index is Count - 1.
For an alternative to visit all children see _NewEnum.

Errors
1700 The object is no longer valid.

Examples
See ProjectItem or Project.NewEnum.

17.2.18.16 Kind

Property: Kind as ENUM ProjectItemType (read-only)

Description
Retrieves the kind of the project item. Availability of some properties and the applicability of certain methods is restricted to specific kinds of project items. The description of all methods and properties contains information about these restrictions.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.17 Name

Property: Name as String

Description
Retrieves or sets the name of a project item. The name of most items is read-only. Exceptions are user-created folders, the names of which can be altered after creation.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 Project item does not allow to alter its name.

17.2.18.18 Open

Method: Open () as Document

Description
Opens the project item as a document or makes the corresponding document the active one, if it is already open. The project item must be a MapForce mapping or, for Enterprise edition only, Web service operation.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 The project item does not refer to a MapForce mapping file.
1708 Operation not supported in current edition.

17.2.18.19 Parent

Property: Parent as Project (read-only)

Description
Retrieves the project that this item is a child of. Has the same effect as Application.ActiveProject.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.20 QualifiedName

Property: QualifiedName as String (read-only)

Description
Retrieves the qualified name of a Web service item.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 The project item is not a part of a Web service.

17.2.18.21 Remove

Method: Remove ()

Description
Remove this project item and all its children from the project tree.

Errors
1700 The object is no longer valid.

17.2.18.22 WSDLFile

Property: WSDLFile as String (read-only)

Description
Retrieves the file name of the WSDL file defining the Web service that hosts the current project item.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 The project item is not a part of a Web service.

17.3 Enumerations

This is a list of all enumerations used by the MapForce API. If your scripting environment does not support enumerations, use the number-values instead.

17.3.1 ENUMApacheAxisVersion (obsolete)

Description
Enumeration values to select the Apache Axis version.

Possible values:

eApacheAxisVersion_Axis	= 1
eApacheAxisVersion_Axis2	= 2

17.3.2 ENUMApplicationStatus

Description
Enumeration values to indicate the status of the application.

Possible values:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3

17.3.3 ENUMAppOutputLine_Severity

Description
Enumeration values to identify the severity of an AppOutputLine.

Possible values:

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

17.3.4 ENUMAppOutputLine_TextDecoration

Description
Enumeration values for the different kinds of text decoration of an AppOutputLine.

Possible values:

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

17.3.5 ENUMCodeGenErrorLevel

Description
Enumeration values to identify severity of code generation messages.

Possible values:

eCodeGenErrorLevel_Information	= 0
eCodeGenErrorLevel_Warning	= 1
eCodeGenErrorLevel_Error	= 2
eCodeGenErrorLevel_Undefined	= 3

17.3.6 ENUMComponentDatapointSide

Description
Enumeration values to indicate the side of a datapoint on its component.

Possible values:

eDatapointSideInput	= 0
eDatapointSideOutput	= 1

See also
GetRootDatapoint

17.3.7 ENUMComponentSubType

Description
Enumeration values to indicate component sub types.

Possible values:

eComponentSubType_None = 0
eComponentSubType_Text_EDI = 1
eComponentSubType_Text_Flex = 2
eComponentSubType_Text_CSVF = 3
LF

17.3.8 ENUMComponentType

Description
Enumeration values to indicate component types.

Possible values:

eComponentType_Unknown = 0
eComponentType_XML = 1
eComponentType_DB = 2
eComponentType_Text = 3
eComponentType_Excel = 4
eComponentType_WSDL = 5
eComponentType_XBRL = 6

17.3.9 ENUMComponentUsageKind

Description
Enumeration values to indicate component usage kind.

Possible values:

eComponentUsageKind_Unknown = 0
eComponentUsageKind_Instance = 1
eComponentUsageKind_Input = 2
eComponentUsageKind_Output = 3
eComponentUsageKind_Variable = 4

17.3.10 ENUMConnectionType

Description
Enumeration values to indicate the type of a connection.

Possible values:

eConnectionTypeTargetDrive = 0

n
eConnectionTypeSourceDriv = 1
en
eConnectionTypeCopyAll = 2

See also
ConnectionType

17.3.11 ENUMDOMType

Description
Enumeration values to specify the DOM type used by generated C++ mapping code.

Possible values:

eDOMType_xerces = 1
eDOMType_xerces3 = 2
eDOMType_msxml6 = 3

Obsolete values

eDOMType_msxml4 = 0

Obsolete in this context means that this value is not supported and should not be used.

`eDOMType_xerces` indicates Xerces 2.x usage; `eDOMType_xerces3` indicates Xerces 3.x usage.

17.3.12 ENUMLibType

Description
Enumeration values to specify the library type used by the generated C++ mapping code.

Possible values:

eLibType_static = 0
eLibType_dll = 1

17.3.13 ENUMProgrammingLanguage

Description
Enumeration values to select a programming language.

Possible values:

eUndefinedLanguage = -1
eJava = 0
eCpp = 1
eCSharp = 2
eXSLT = 3
eXSLT2 = 4
eXQuery = 5

17.3.14 ENUMProjectItemType

WDescription
Enumeration to identify the different kinds of project items that can be children of Project or folder-like ProjectItems.

Possible values:

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_LibrarzFolder	= 8
eProjectItemType_ResourceFolder	= 9
eProjectItemType_VirtualFolder	= 10

See also
ProjectItem Kind

17.3.15 ENUMProjectType

Description
Enumeration values to select a project type for generated C# mapping code.

Possible values:

	eVisualStudio2005Project	= 4 Also for C++ code
	eVisualStudio2008Project	= 5 Also for C++ code
	eVisualStudio2010Project	= 6 Also for C++ code
Obsolete values	eVisualStudioProject	= 0
	eVisualStudio2003Project	= 1
	eBorlandProject	= 2

Obsolete in this context means that this value is not supported and should not be used.

17.3.16 ENUMSearchDatapointFlags

Description
Enumeration values used as bit-flags; to be used as combination of flags when searching for a datapoint.

Possible values:

eSearchDatapointElement	= 1
eSearchDatapointAttribute	= 2

See also
GetChild

17.3.17 ENUMViewMode

Description
Enumeration values to select a MapForce view.

Possible values:

eMapForceView	= 0
eXSLView	= 1
eOutputView	= 2

Chapter 18

ActiveX Integration

18 ActiveX Integration

The MapForce user interface and the functionality described in this section can be integrated into custom applications that can consume ActiveX controls. ActiveX technology enables a wide variety of languages to be used for integration, such as C++, C#, VB.NET, HTML. (Note that ActiveX components integrated in HTML must be run with Microsoft Internet Explorer versions and platforms that support ActiveX). All components are full OLE Controls. Integration into Java is provided through wrapper classes.

> To integrate the ActiveX controls into your custom code, the MapForce Integration Package must be installed (see http://www.altova.com/ide_integration.html). Ensure that you install MapForce first, and then the MapForce Integration Package. Other prerequisites apply, depending on language and platform (see Prerequisites).

You can flexibly choose between two different levels of integration: application level and document level.

Integration at application level means embedding the complete interface of MapForce (including its menus, toolbars, panes, etc) as an ActiveX control into your custom application. For example, in the most simple scenario, your custom application could consist of only one form that embeds the MapForce graphical user interface. This approach is easier to implement than integration at document level but may not be suitable if you need flexibility to configure the MapForce graphical user interface according to your custom requirements.

Integration at document level means embedding MapForce into your own application piece-by-piece. This includes implementing not only the main MapForce control but also the main document editor window, and, optionally, any additional windows. This approach provides greater flexibility to configure the GUI, but requires advanced interaction with ActiveX controls in your language of choice.

The sections Integration at the Application Level and Integration at Document Level describe the key steps at these respective levels. The ActiveX Integration Examples section provides examples in C#, HTML, and Java. Looking through these examples will help you to make the right decisions quickly. The Object Reference section describes all COM objects that can be used for integration, together with their properties and methods.

For information about using MapForce as a Visual Studio plug-in, see MapForce in Visual Studio.

(C) 2017 Altova GmbH

18.1 Prerequisites

To integrate the MapForce ActiveX control into a custom application, the following must be installed on your computer:

- MapForce
- The MapForce Integration Package, available for download at http://www.altova.com/ide_integration.html

To integrate the 64-bit ActiveX control, install the 64-bit versions of MapForce and MapForce Integration Package. For applications developed under Microsoft .NET platform with Visual Studio, both the 32-bit and 64-bit versions of MapForce and MapForce Integration Package must be installed, as explained below.

Note: For the integration package to function correctly on Windows XP and Windows Server 2003, you will need to manually install the Windows redistributable available at https://www.microsoft.com/en-US/download/details.aspx?id=48145.

Microsoft .NET (C#, VB.NET) with Visual Studio

To integrate the MapForce ActiveX control into a 32-bit application developed under Microsoft .NET, the following must be installed on your computer:

- Microsoft .NET Framework 4.0 or later
- Visual Studio 2005/2008/2010/2012/2013/2015/2017
- MapForce 32-bit and MapForce Integration Package 32-bit
- The ActiveX controls must be added to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

If you want to integrate the 64-bit ActiveX control, the following prerequisites apply in addition to the ones above:

- MapForce 32-bit and MapForce Integration Package 32-bit must still be installed (this is required to provide the 32-bit ActiveX control to the Visual Studio designer, since Visual Studio runs on 32-bit)
- MapForce 64-bit and MapForce Integration Package 64-bit must be installed (provides the actual 64-bit ActiveX control to your custom application at runtime)
- In Visual Studio, create a 64-bit build configuration and build your application using this configuration. For an example, see Running the Sample C# Solution.

Java

To integrate the MapForce ActiveX control into Java application using the Eclipse development environment, the following must be installed on your computer:

- Java Runtime Environment (JRE) or Java Development Kit (JDK) 7 or later
- Eclipse
- MapForce and MapForce Integration Package

Note: To run the 64-bit version of the MapForce ActiveX control, use a 64-bit version of Eclipse, as well as the 64-bit version of MapForce and the MapForce Integration Package.

MapForce integration and deployment on client computers

If you create a .NET application and intend to distribute it to other clients, you will need to install the following on the client computer(s):

- MapForce
- The MapForce Integration Package
- The custom integration code or application.

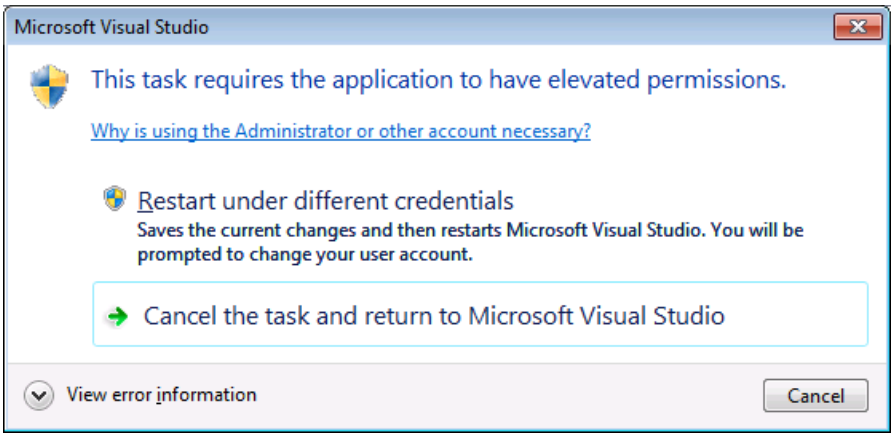
18.2 Adding the ActiveX Controls to the Toolbox

To use the MapForce ActiveX controls in an application developed with Visual Studio, the controls must first be added to the Visual Studio Toolbox, as follows:

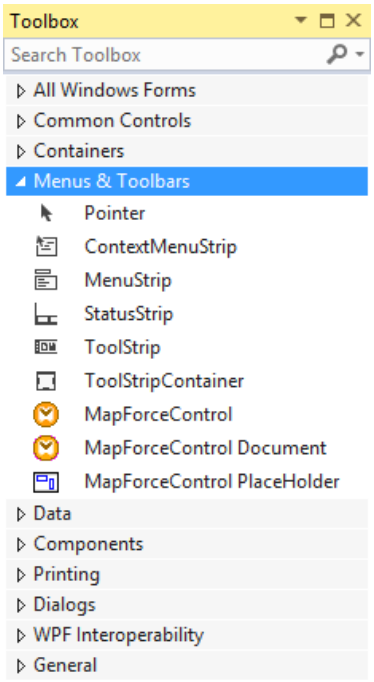
1. On the **Tools** menu of Visual Studio, click **Choose Toolbox Items**.
2. On the **COM Components** tab, select the check boxes next to the MapForceControl, MapForceControl Document, and MapForceControl Placeholder.

In case the controls above are not available, follow the steps below:

1. On the **COM Components** tab, click **Browse**, and select the **MapForceControl.ocx** file from the MapForce installation folder. Remember that the MapForce Integration Package must be installed; otherwise, this file is not available, see Prerequisites.
2. If prompted to restart Visual Studio with elevated permissions, click **Restart under different credentials**.



If the steps above were successful, the MapForce ActiveX controls become available in the Visual Studio Toolbox.



Note: For an application-level integration, only the **MapForceControl** ActiveX control is used (see Integration at Application Level). The **MapForceControl Document** and **MapForceControl Placeholder** controls are used for document-level integration (see Integration at Document Level).

18.3 Integration at Application Level

Integration at application level allows you to embed the complete interface of MapForce into a window of your application. With this type of integration, you get the whole user interface of MapForce, including all menus, toolbars, the status bar, document windows, and helper windows. Customization of the application's user interface is restricted to what MapForce provides. This includes rearrangement and resizing of helper windows and customization of menus and toolbars.

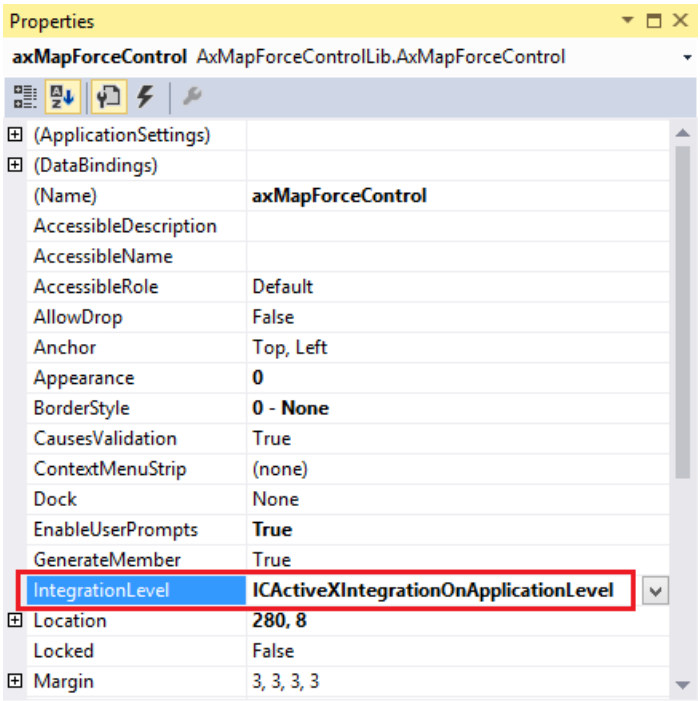
The only ActiveX control you need to integrate is `MapForceControl`. Do not instantiate or access `MapForceControlDocument` or `MapForceControlPlaceHolder` ActiveX controls when integrating at application-level.

If you have any initialization to do or if you want to automate some behaviour of MapForce, use the properties, methods, and events described for `MapForceControl`. Consider using `MapForceControl.Application` for more complex access to MapForce functionality.

For an example that shows how the MapForce application can be embedded in an HTML page, see HTML Integration at Application Level.

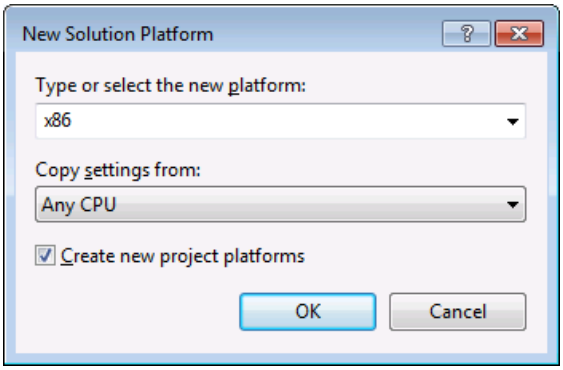
In C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at application level are as follows:

1. Check that all prerequisites are met (see Prerequisites).
2. Create a new Visual Studio Windows Forms project with a new empty form.
3. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
4. Drag the **MapForceControl** from the toolbox onto your new form.
5. Select the **MapForceControl** on the form, and, in the Properties window, set the **IntegrationLevel** property to **ICActiveXIntegrationOnApplicationLevel**.



6. Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

   a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
   b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64).

(C) 2017 Altova GmbH

18.4 Integration at Document Level

Compared to integration at application level, integration at document level is a more complex, yet more flexible way to embed MapForce functionality into your application by means of ActiveX controls. With this approach, your code can access selectively the following parts of the MapForce user interface:

- Document editing window
- Project window
- Libraries window
- Overview window
- Messages window

As mentioned in Integration at Application Level, for an ActiveX integration at application level, only one control is required, namely the **MapForceControl**. However, for an ActiveX integration at document level, functionality MapForce is provided by the following ActiveX controls:

1. **MapForceControl**
2. **MapForceControl Document**
3. **MapForceControl Placeholder**

These controls are supplied by the **MapForceControl.ocx** file available in the application installation folder of MapForce. When you develop the ActiveX integration with Visual Studio, you will need to add these controls to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

The basic steps to integrate the ActiveX controls at document level into your application are as follows:

1. First, instantiate **MapForceControl** in your application. Instantiating this control is mandatory; it enables support for the **MapForceControl Document** and **MapForceControl Placeholder** controls mentioned above. It is important to set the `IntegrationLevel` property to **ICActiveXIntegrationOnDocumentLevel** (or "1"). To hide the control from the user, set its **Visible** property to **False**.

Note: When integrating at document level, do not use the **Open** method of the **MapForceControl**; this might lead to unexpected results. Use the corresponding open methods of **MapForceControl Document** and **MapForceControl PlaceHolder** instead.

2. Create at least one instance of **MapForceControl Document** in your application. This control supplies the document editing window of MapForce to your application and can be instantiated multiple times if necessary.

   Use the method **Open** to load any existing file. To access document-related functionality, use the **Path** and **Save** or methods and properties accessible via the property **Document**.

Note: The control does not support a read-only mode. The value of the property **ReadOnly** is ignored.

3. Optionally, add to your application the **MapForceControl Placeholder** control for each additional window (other than the document window) that must be available to your application.

   Instances of **MapForceControl PlaceHolder** allow you to selectively embed additional

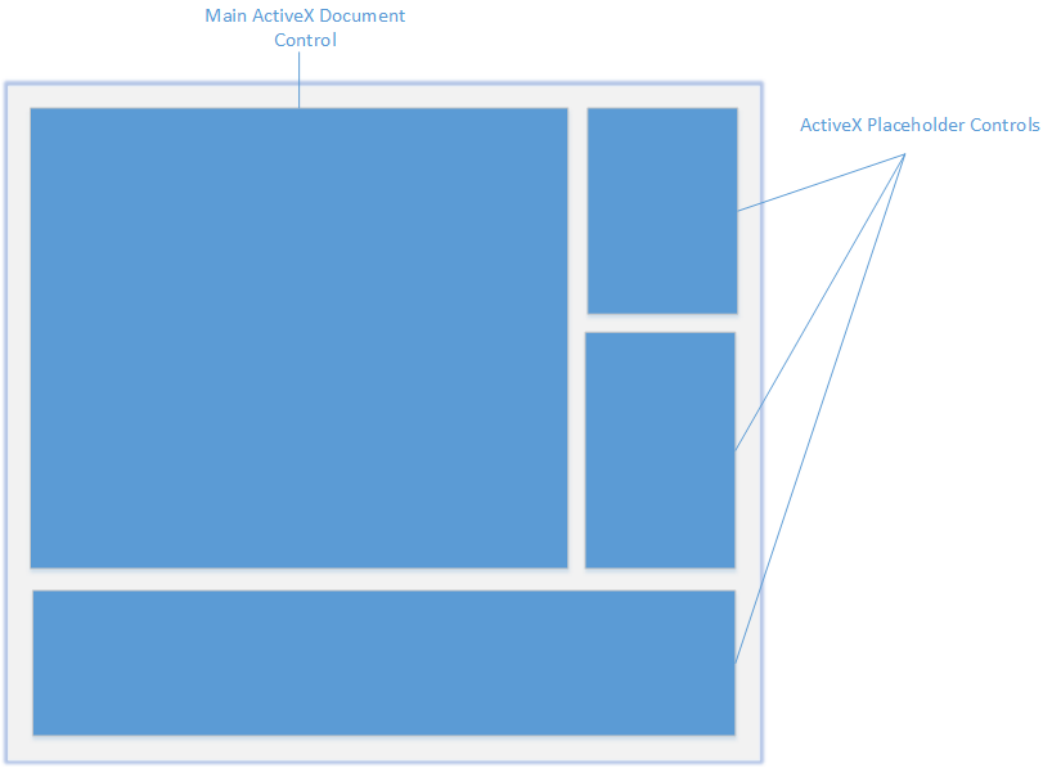
windows of MapForce into your application. The window kind (for example, Project window) is defined by the property **PlaceholderWindowID**. Therefore, to set the window kind, set the property **PlaceholderWindowID**. For valid window identifiers, see MapForceControlPlaceholderWindow.

Note: Use only one **MapForceControl PlaceHolder** for each window identifier.

For placeholder controls that select the MapForce project window, additional methods are available. Use **OpenProject** to load a MapForce project. Use the property Project and the methods and properties from the MapForce automation interface to perform any other project related operations.

For example, in C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at document level could be similar to those listed below. Note that your application may be more complex if necessary; however, the instructions below are important to understand the minimum requirements for an ActiveX integration at document level.

1. Create a new Visual Studio Windows Forms project with a new empty form.
2. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
3. Drag the **MapForceControl** from the toolbox onto your new form.
4. Set the **IntegrationLevel** property of the **MapForceControl** to **ICActiveXIntegrationOnDocumentLevel**, and the **Visible** property to **False**. You can do this either from code or from the **Properties** window.
5. Drag the **MapForceControl Document** from the toolbox onto the form. This control provides the main document window of MapForce to your application, so you may need to resize it to a reasonable size for a document.
6. Optionally, add one or more **MapForceControl Placeholder** controls to the form (one for each additional window type that your application needs, for example, the **Project** window). You will typically want to place such additional placeholder controls either below or to the right or left of the main document control, for example:



7. Set the **PlaceholderWindowID** property of each **MapForceControl Placeholder** control to a valid window identifier. For the list of valid values, see MapForceControlPlaceholderWindow.
8. Add commands to your application (at minimum, you will need to open, save and close documents), as shown below.

Querying MapForce Commands

When you integrate at document level, no MapForce menu or toolbar is available to your application. Instead, you can retrieve the required commands, view their status, and execute them programmatically, as follows:

- To retrieve all available commands, use the `CommandsList` property of the **MapForceControl**.
- To retrieve commands organized according to their menu structure, use the `MainMenu` property.
- To retrieve commands organized by the toolbar in which they appear, use the `Toolbars` property.
- To send commands to MapForce, use the `Exec` method.
- To query if a command is currently enabled or disabled, use the `QueryStatus` method.

This enables you to flexibly integrate MapForce commands into your application's menus and toolbars.

Your installation of MapForce also provides you with command label images used within MapForce. See the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your MapForce installation for icons in GIF format. The file names correspond to the command names as they are listed in the Command Reference section.

General considerations

To automate the behaviour of MapForce, use the properties, methods, and events described for the **MapForceControl**, **MapForceControl Document**, and **MapForceControl Placeholder**.

For more complex access to MapForce functionality, consider using the following properties:

- `MapForceControl.Application`
- `MapForceControlDocument.Document`
- `MapForceControlPlaceHolder.Project`

These properties give you access to the MapForce automation interface (MapForceAPI)

Note: To open a document, always use `MapForceControlDocument.Open` or `MapForceControlDocument.New` on the appropriate document control. To open a project, always use `MapForceControlPlaceHolder.OpenProject` on a placeholder control embedding a MapForce project window.

For examples that show how to instantiate and access the necessary controls in different programming environments, see ActiveX Integration Examples.

18.5 ActiveX Integration Examples

This section contains examples of MapForce document-level integration using different container environments and programming languages. (The HTML section additionally contains examples of integration at application level.) Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX` of your MapForce installation.

18.5.1 C#

A basic ActiveX integration example solution for C# and Visual Studio is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#`. Before you compile the source code and run the sample, make sure that all prerequisites are met (see Running the Sample C# Solution).

18.5.1.1 Running the Sample C# Solution

The sample Visual Studio solution available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** illustrates how to consume the MapForce ActiveX controls. Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

Visual Studio 2010 or later is required to open the sample solution. For the complete list of prerequisites, see Prerequisites.

Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Check and set all required control properties

The sample application contains one instance of MapForceControlDocument and several instances of MapForceControlPlaceHolder controls. Double-check that the following properties of these controls are set as shown in the table below:

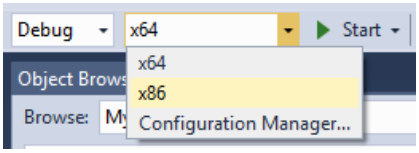
Control name	Property	Property value
axMapForceControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
axMapForceControlLibrary	PlaceholderWindowID	0
axMapForceControlOutput	PlaceholderWindowID	2

Control name	Property	Property value
axMapForceControlPreview	PlaceholderWindowID	1

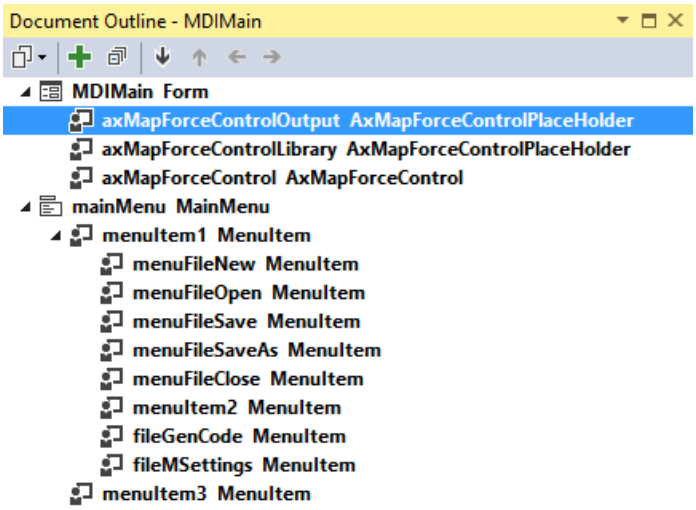
Here is how you can view or set the properties of an ActiveX control:

1. Open the **MDIMain.cs** form in the designer window.

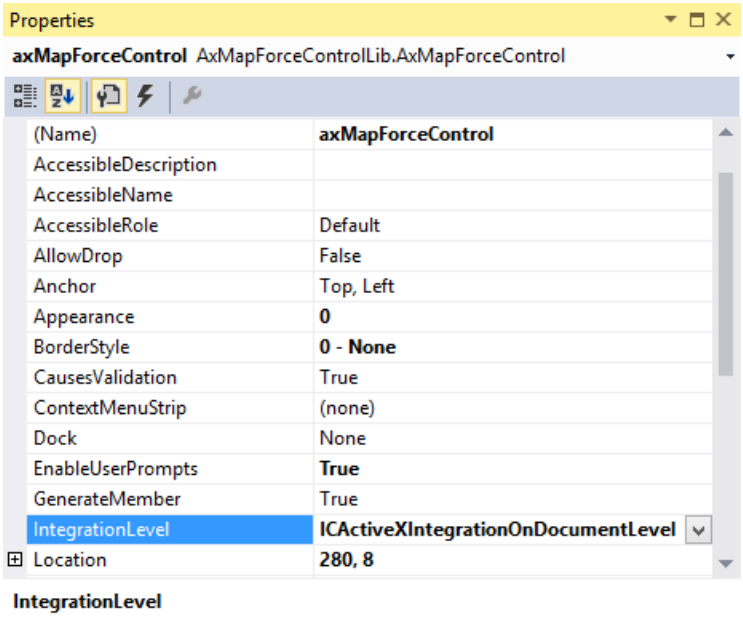
Note: On 64-bit Windows, it may be necessary to change the build configuration of the Visual Studio solution to "x86" **before** opening the designer window. If you need to build the sample as a 64-bit application, see Prerequisites.



2. Open the **Document Outline** window of Visual Studio (On the **View** menu, click **Other Windows | Document Outline**).



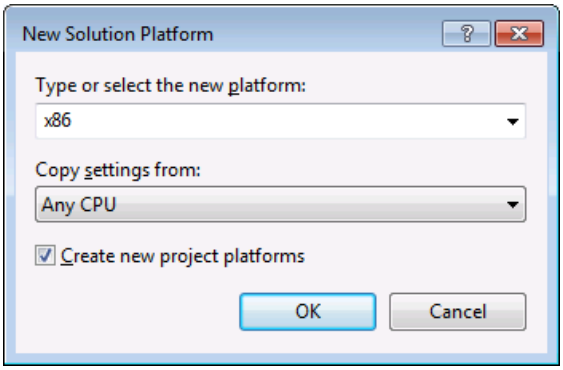
3. Click an ActiveX control in the **Document Outline** window, and edit its required property in the **Properties** window, for example:



Step 4: Set the build platform

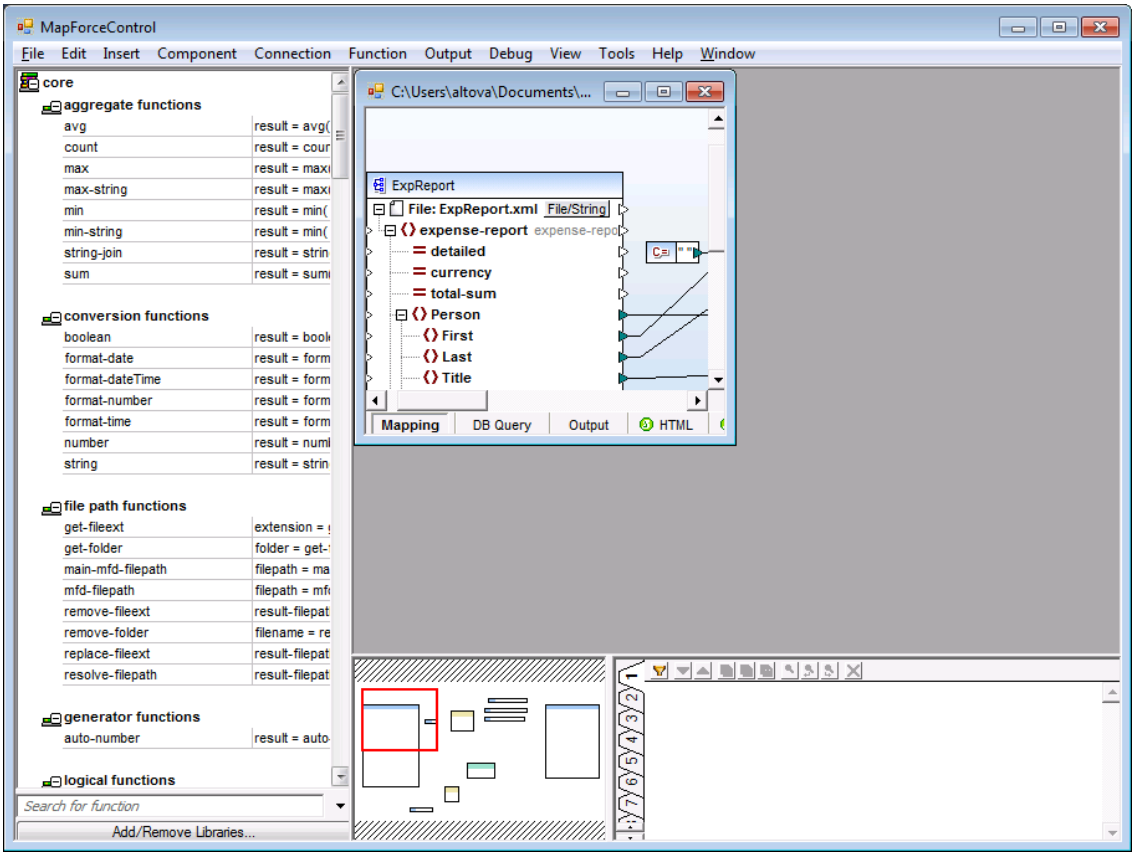
- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

On running the sample, the main MDI Frame window is displayed. Use **File | Open** to open a mapping file (for example, `MarketingExpenses.mfd`, which is in the MapForce examples folder). The file is loaded and displayed in a new document child window:



Click to enlarge image

After you load the document, you can execute commands against the active document using the menu. Context menus are also available. You can also load additional documents. Save any modifications using the **File | Save** command.

18.5.1.2 Retrieving Command Information

The MapForceControl gives access to all commands of MapForce through its `CommandsList` , `MainMenu`, and `Toolbars` properties. The example project available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** uses the `MainMenu` property to create the MapForce menu structure dynamically.

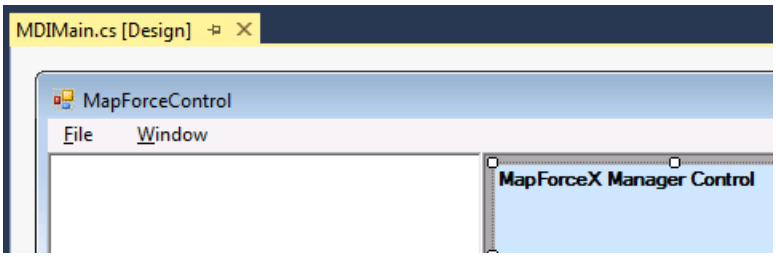
The code that gets the menu commands can be found in the `MDIMain` method in `MDIMain.cs` file:

```
public MDIMain()
{
    // ...

    // Get the MainMenu property of the control and create the menu
structure from it.
    MFLib.MapForceCommand objCommand = this.axMapForceControl.MainMenu;
    InsertMenuStructure(mainMenu, objCommand);
}
```

In the code listing above, `mainMenu` is the existing static menu of the main MDI Frame window. If you open the **MDIMain.cs** form in the Visual Studio Designer, you will notice that this menu

contains two menu items: **File** and **Window**.



MDIMain.cs

The method `InsertMenuStructure` takes as parameters the `mainMenu` and the `objCommand` objects (the former is the existing static menu, while the latter contains the full menu structure retrieved from the MapForce ActiveX control). The retrieved MapForce menu structure is then merged into the existing static menu. Note that the menus **File**, **Project**, and **Window** are not added dynamically. This is intentional, because these menus deal with actively open documents, and they would require code which is beyond the scope of this example. The basic file management commands (create, open, save, bring into focus) are handled by the existing static menus **File** and **Window**. All other menus are inserted dynamically based on the information taken from the `MainMenu` property of the ActiveX control. The new menus are inserted after "File" but before "Window", i.e. starting at menu index 1.

The method `InsertMenuStructure` iterates through all top-level menus found in `MapForceCommand` object and adds a new menu item for each. Since each top-level menu has its own child menu items, a call to the method `InsertMenuCommand` takes place for each encountered child menu item. Furthermore, since each child menu item can have its own children menu items, and so on, the `InsertMenuCommand` method recurses into itself until no more child menu items exist.

The commands added dynamically are instances of the class `CustomMenuItem`, which is defined in `CustomMenuItem.cs`. This class is derived from `System.Windows.Forms.MenuItem` class and has an additional member to store the MapForce command ID.

```
public class CustomMenuItem :System.Windows.Forms.MenuItem
{
	public int m_MapForceCmdID;
}
```

All dynamically added commands (except those that are containers for other 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("MapForceApplication.CustomMenuItem"))
  {
    CustomMenuItem   customItem = (CustomMenuItem)sender;
    ProcessCommand(customItem.m_MapForceCmdID);
  }
}
```

If the command is a container for other commands (that is, if it has child commands), it gets the event handler `AltovaSubMenu_Popup`. This handler queries the status of each child command and enables or disables it as required. This ensures that each command is enabled only when

that is meaningful (for example, the **File | Save** menu item should be disabled if there is no active document open).

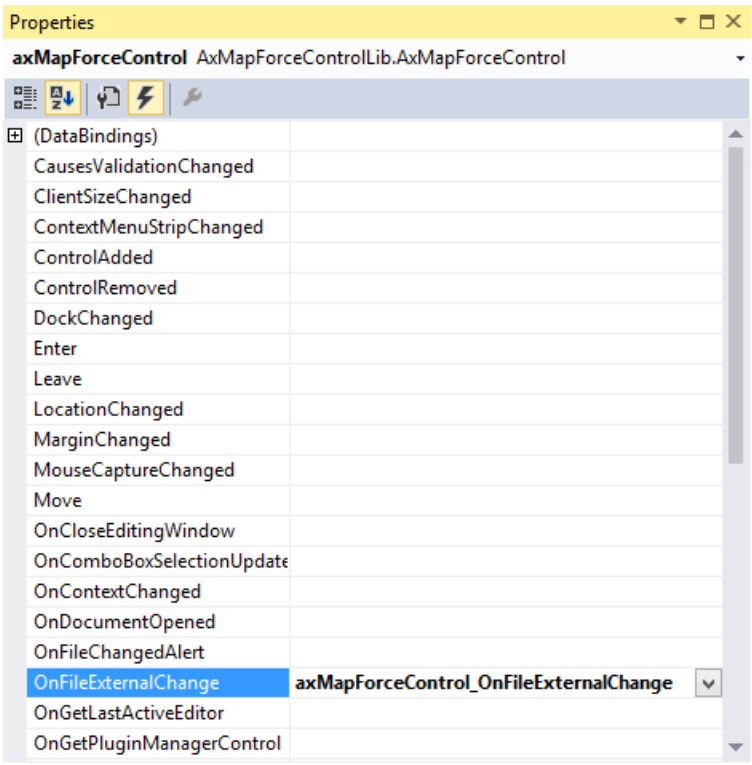
The method `ProcessCommand` delegates the execution either to the MapForceControl itself or to any active MapForce document loaded in a `MapForceControlDocument` control. This is necessary because the MapForceControl has no way to know which document is currently active in the hosting application.

```
private void ProcessCommand(int nID)
{
   MapForceDoc docMapForce = GetCurrentMapForceDoc();

   if(docMapForce != null)
      docMapForce.axMapForceControlDoc.Exec(nID);
   else
      axMapForceControl.Exec(nID);
}
```

18.5.1.3 Handling Events

Because all events in the MapForce library are based on connection points, you can use the C# delegate mechanism to provide the custom event handlers. You will always find a complete list of events on the property page of each control of the MapForce library. The image below shows the events of the main MapForceControl:



As you can see, the example project only overrides the `OnFileExternalChange` event. The creation of the C# delegate is done for you by the C# Framework. All you need to do is fill in the empty event handler.

For example, the handler implementation shown below turns off any file reloading and displays a message box to inform the user that a file loaded by the MapForceControl has been changed from outside:

```
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;
}
```

18.5.2 HTML

The code listings in this section show how to integrate the MapForceControl at application level and document level. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\HTML` of your MapForce installation. The examples work only in Internet Explorer when it runs as a 32-bit application.

Note: When it runs in 64-bit mode, Internet Explorer 10 or later does not load ActiveX controls.

18.5.2.1 HTML Integration at Application Level

This example shows a simple integration of the MapForce control at application-level into a HTML page. The integration is described in the following sections:

- Instantiate a MapForceControl in HTML code.
- Implement buttons to load documents and automate code-generation tasks.
- Define actions for some application events.

The code for this example is available at the following location in your MapForce installation: `<ApplicationFolder>\Examples\ActiveX\HTML\MapForceActiveX_ApplicationLevel.htm`.

Instantiate the Control

The HTML `Object` tag is used to create an instance of the MapForceControl. The `Classid` is that of MapForceControl. Width and height specify the window size. No additional parameters are necessary, since application-level is the default.

```
<OBJECT id="objMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="800"
        height="500"
        VIEWASTEXT>
</OBJECT>
```

Add Button to Open Default Document

As a simple example of how to automate some tasks, we add a button to the page:

```
<input type="button" value="Open Marketing Expenses"
onclick="BtnOpenMEFile()">
```

When clicked, a predefined document will be opened in the MapForceControl. We use a method to locate the file relative to the MapForceControl so the example can run on different installations.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// --------------------------------
// open a pre-defined document
function BtnOpenMEFile()
{

            objMapForceControl.Open("C:\Documents and Settings\username\My
Documents\Altova\XMLSpy2017\Examples/MarketingExpenses.mfd");

}
</SCRIPT>
```

Add Buttons for Code Generation

Although code-generation for the active document is available via menus, we want to have buttons that will generate code without asking the user for the location of the output. The method is similar to that used in the previous section.

First come the buttons:

```
<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 )">
```

Then we provide the script that will generate the code into sub-folders of the currently defined default output folders.

```
<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>
```

Connect to Custom Events

The example implements two event callbacks for MapForceControl custom events to show the principle:

```
<!-- ------------------------------------------------------------ -->
<!--  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>
```

18.5.2.2 HTML Integration at Document Level

This example shows an integration of the MapForce control at document-level into a HTML page. The following topics are covered:

- Instantiate a MapForceControl ActiveX control object in HTML code
- Instantiate a MapForceControlDocument ActiveX control to allow editing a MapForce file
- Instantiate one MapForceControlPlaceHolder for a MapForceControl project window
- Instantiate one MapForceControlPlaceHolder to alternatively host one of the MapForce helper windows
- Create a simple customer toolbar for some heavy-used MapForce commands
- Add some more buttons that use the COM automation interface of MapForce
- Use event handlers to update command buttons

This example is available in its entirety in the file `MapForceActiveX_ApplicationLevel.htm` within the `<ApplicationFolder>\Examples\ActiveX\HTML\` folder of your MapForce installation.

Instantiate the MapForceControl

MapForceControlThe HTML `OBJECT` tag is used to create an instance of the MapForceControl. The Classid is that of MapForceControl. Width and height are set to 0 since we use this control as manager control without use for its user interface. The integration level is specified as a parameter within the `OBJECT` tag.

```
<OBJECT id="objMapForceXMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="0"
        height="0"
        VIEWASTEXT>
   <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

Create Editor Window

The HTML `OBJECT` tag is used to embed an editing window. The additional custom parameter specifies that the control is to be initialized with a new empty mapping.

```
<OBJECT id="objDoc1"
        Classid="clsid:DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
        width="600"
        height="500"
        VIEWASTEXT>
    <PARAM NAME="NewDocument">
</OBJECT>
```

Create Project Window

The HTML `OBJECT` tag is used to create a MapForceControlPlaceHolder window. The first additional custom parameter defines the placeholder to show the MapForce project window. The second parameter loads one of the example projects delivered with your MapForce installation (located in the `<yourusername>/MyDocuments` folder).

```
<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>
```

Create Placeholder for Helper Windows

The HTML OBJECT tag is used to instantiate a MapForceControlPlaceHolder ActiveX control that can host the different MapForce helper windows. Initially, no helper window is shown. See the example file.

```
<OBJECT id="objEHWindow"
        Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
        width="200"
        height="200"
        VIEWASTEXT>
       <PARAM name="PlaceholderWindowID" value="0">
</OBJECT>
```

Three buttons allow us to switch the actual window that will be shown. The JavaScript execute on-button-click sets the property PlaceHolderWindowID to the corresponding value defined in

```
<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>
```

Create a Custom Toolbar

The custom toolbar consists of buttons with images of MapForce commands. The command ID numbers can be found in the button elements shown below

```
<button id="btnInsertXML" title="Insert XML Schema/File"
onclick="BtnDoCommand(13635)">
         <img src="..\Images\ID_INSERT_XSD.gif" width="16" height="16" />
</button>
<button id="btnInsertDB" title="Insert Database"
onclick="BtnDoCommand(13590)">
         <img src="..\Images\ID_INSERT_DATABASE.gif" width="16" height="16" /
>
```

```
</button>
<button id="btnInsertEDI" title="Insert EDI" onclick="BtnDoCommand(13591)">
          <img src="..\Images\ID_INSERT_EDI.gif" width="16" height="16" />
</button>
```

On clicking one of these buttons the corresponding command ID is sent to the manager control.

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ---------------------------------------------------------------
// perform any command specified by cmdID.
// command routing includes application, active document and view.
function BtnDoCommand(cmdID)
{
      objMapForceXExec(cmdID);
      msgtext.innerText = "Command " + cmdID + " performed.";
}
</SCRIPT>
```

Create More Buttons

In the example, we add some more buttons to show some automation code.

```
<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>
```

The corresponding JavaScript looks like this:

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ---------------------------------------------------------
// open a document in the specified document control window.
function BtnOpenMEFile(objDocCtrl)
{
      // do not use MapForceXApplicationOpenDocument(…) to open a document,
      // since then MapForceControl wouldn't know a control window to show
      // the document in. Instead:

      objDocCtrlOpenDocument("C:\Documents and Settings\username\My Documents\
            Altova\XMLSpy2017\Examples/MarketingExpenses.mfd");
      objDocCtrlsetActive();

}

// ---------------------------------------------------------------
```

```
// 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>
```

Create Event Handler to Update Button Status

Availability of a command may vary with every mouse click or keystroke. The custom event OnUpdateCmdUI of MapForceControl gives us an opportunity to update the enabled/disabled state of buttons associated with MapForce commands. The method MapForceControl.QueryStatus is used to query whether a command is enabled or not.

```
<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>
```

18.5.3 Java

MapForce ActiveX components can be accessed from Java code. Java integration is provided by the libraries listed below. These libraries are available in the folder `<ApplicationFolder>\Examples\JavaAPI` of your MapForce installation, after you have installed both MapForce and the MapForce Integration Package (see also Prerequisites).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (in case of the 32-bit installation of MapForce)
- `AltovaAutomation_x64.dll`: a JNI wrapper for Altova automation servers (in case of the 64-bit installation of MapForce)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceActiveX.jar`: Java classes that wrap the MapForce ActiveX interface
- `MapForceActiveX_JavaDoc.zip`: a Javadoc file containing help documentation for the Java interface

Note: In order to use the Java ActiveX integration, the .dll and .jar files must be included in the Java class search path.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details, see Example Java Project.

Rules for mapping the ActiveX Control names to Java

For the documentation of ActiveX controls, see Object Reference. Note that the object naming conventions are slightly different in Java compared to other languages. Namely, the rules for mapping between the ActiveX controls and the Java wrapper are as follows:

- ***Classes and class names***
  For every component of the MapForce ActiveX interface a Java class exists with the name of the component.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the MapForceControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

- ***Enumerations***
  For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `MapForceControl`: Java class to access the application
  `MapForceControlEvents`: Events interface for the MapForceControl
  `MapForceControlEventsDefaultHandler`: Default handler for `MapForceControlEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Changes in Java class
`MapForceControlDocument`, method `New`	Renamed to `newDocument`
`MapForceControlDocument`, method `OpenDocument`	Removed. Use the `Open` method
`MapForceControlDocument`, method `NewDocument`	Removed. Use the `newDocument` method
`MapForceControlDocument`, method `SaveDocument`	Removed. Use the `Save` method

This section

This section shows how some basic MapForce ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project
- Creating the ActiveX Controls
- Loading Data in the Controls
- Basic Event Handling
- Menus
- UI Update Event Handling
- Creating a MapForce Mapping Table

18.5.3.1 Example Java Project

The MapForce installation package contains an example Java project, located in the ActiveX Examples folder of the application folder: `<ApplicationFolder>\Examples\ActiveX\Java\`.

The Java example shows how to integrate the MapForceControl in a common desktop application created with Java. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`AltovaAutomation.dll`	Java-COM bridge: DLL part (for the 32-bit installation)
`AltovaAutomation_x64.dll`	Java-COM bridge: DLL part (for the 64-bit installation)
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`MapForceActiveX.jar`	Java classes of the MapForce ActiveX control
`MapForceActiveX_JavaDoc.zip`	Javadoc file containing help documentation for the Java API
`MapForceContainer.java`	Java example source code
`MapForceContainerEventHandler.java`	Java example source code
`MapForceTable.java`	Java example source code

What the example does

The example places one MapForce document editor window, the MapForce project window, the MapForce library window and the MapForce validation window in an AWT frame window. It reads out the main menu defined for MapForce and creates an AWT menu with the same structure. You can use this menu or the project window to open and work with files in the document editor.

You can modify the example in any way you like.

The following specific features are described in code listings:

- Creating the ActiveX Controls: Starts MapForce, which is registered as an automation server, or activates MapForce if it is already running.
- Loading Data in the Controls: Locates one of the example documents installed with MapForce and opens it.
- Basic Event Handling: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- UI Update Event Handling: Shows how to handle MapForce events.
- Creating a MapForce Mapping Table: Shows how to create a MapForce mapping table and prepare it for modal activation.

Updating the path to the Examples folder

Before running the provided sample, you may need to edit the **MapForceContainer.java** file. Namely, check that the following path refers to the actual folder where the MapForce example files are stored on your operating system:

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\My
Documents\\Altova\\MapForce2017\\MapForceExamples\\";
```

Running the example from the command line

To run the example from the command line:

1. Check that all prerequisites are met (see Prerequisites).
2. Open a command prompt window, change the current directory to the sample Java project folder, and type:

   ```
   buildAndRun.bat "<Path-to-the-Java-bin-folder>"
   ```

3. Press **Enter**.

The Java source in `MapForceContainer.java` will be compiled and then executed.

Compiling and running the example in Eclipse

To import the sample Java project into Eclipse:

1. Check that all prerequisites are met (see Prerequisites).
2. On the **File** menu, click **Import**.
3. Select **Existing Projects into Workspace**, and browse for the Eclipse project file located at `<ApplicationFolder>\Examples\ActiveX\Java\`. Since you may not have write-access in this folder, it is recommended to select the **Copy projects into workspace** check box on the Import dialog box.

To run the example application, right-click the project in Package Explorer and select the command **Run as | Java Application**.

Help for Java API classes is available through comments in code as well as the Javadoc view of Eclipse. To enable the Javadoc view in Eclipse, select the menu command **Window | Show View | JavaDoc**.

18.5.3.2 Creating the ActiveX Controls

The code listing below show how ActiveX controls can be created. The constructors will create the Java wrapper objects. Adding these Canvas-derived objects to a panel or to a frame will trigger the creation of the wrapped ActiveX object.

```
01 /**
02  * MapForce manager control - always needed
03  */
04 public static MapForceControl     mapForceControl= null;
05
06 /**
07  * MapForceDocument editing control
```

```
08   */
09  public static MapForceControlDocument   mapForceDocument = null;
10
11  /**
12   * Toolwindows - MapForce place-holder controls
13   */
14  private static MapForceControlPlaceHolder  mapForceProjectToolWindow = null;
15  private static MapForceControlPlaceHolder  mapForceValidationToolWindow = null;
16  private static MapForceControlPlaceHolder  mapForceLibraryToolWindow = null;
17
18  // Create the MapForce ActiveX control; the parameter determines that we want
    // to place document controls and place-holder controls individually.
19  // It gives us full control over the menu, as well.
20    mapForceControl= new MapForceControl(
      IC ActiveXIntegrationLevel.IC ActiveXIntegrationOnDocumentLevel.getValue(), false );
21
22    mapForceDocument = new MapForceControlDocument();
23    frame.add(mapForceDocument, BorderLayout.CENTER );
24
25
26  // Create a project window and open the sample project in it
27    mapForceProjectToolWindow = new MapForceControlPlaceHolder(
      MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(),
      strExamplesFolder + "MapForceExamples.mfp" );
28    mapForceProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
```

18.5.3.3 Loading Data in the Controls

The code listing below show how data can be loaded in the ActiveX controls.

```
1   // Locate samples installed with the product.
2     final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
      "\\My Documents\\Altova\\MapForce2017\\MapForceExamples\\";
3     mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(), strExamplesFolder
+ "MapForceExamples.mfp" );
```

18.5.3.4 Basic Event Handling

The code listing below shows how basic events can be handled. When calling the MapForceControl's `open` method, or when trying to open a file via the menu or Project tree, the `onOpenedOrFocused` event is sent to the attached event handler. The basic handling for this event is opening the file by calling the MapForceDocumentControl's `open` method.

```
01     // Open the Marketing file when button is pressed
02     btnMarkExp.addActionListener( new ActionListener(){
03       public void actionPerformed(ActionEvent e){
04         try {
05           // Instruct the Document control to open the file - avoid calling the open method of MapForceControl (see
help)
06           mapForceDocument.open( strExamplesFolder + "MarketingExpenses.mfd" );
07           mapForceDocument.requestFocusInWindow();
08         }catch (AutomationException e1){
09           e1.printStackTrace();
10         }
11       }
12     });
13     public void onOpenedOrFocused( String i_strFileName, boolean ib_OpenWithThisControl, boolean
ib_FileAlreadyOpened ) throws AutomationException
14   {
15     // Handle the New/Open events coming from the Project tree or from the menus
16     if ( !ib_FileAlreadyOpened )
17     {
18       // This is basically an SDI interface, so open the file in the already existing document control
19       try {
20         MapForceContainer.mapForceDocument.open( i_strFileName );
```

```
21      MapForceContainer mapForceDocument.requestFocusInWindow();
22    }catch (Exception e){
23      e.printStackTrace();
24    }
25   }
26  }
```

18.5.3.5 Menus

The code listing below shows how menu items can be created. Each `MapForceCommand` object gets a corresponding `MenuItem` object, with the `ActionCommand` set to the ID of the command. The actions generated by all menu items are handled by the same function, which can perform specific handlings (like reinterpreting the closing mechanism) or can delegate the execution to the `MapForceControl` object by calling its `exec` method. The `menuMap` object that is filled during menu creation is used later (see section UI Update Event Handling).

```
01
02    // Load the file menu when the button is pressed
03    btnMenu.addActionListener(new ActionListener(){
04      public void actionPerformed(ActionEvent e){
05        try {
06          // Create the menubar that will be attached to the frame
07          MenuBar mb = new MenuBar();
08          // Load the main menu's first item - the File menu
09          MapForceCommand xmlSpyMenu = mapForceControl.getMainMenu().getSubCommands().getItem(0);
10          // Create Java menu items from the Commands objects
11          Menu fileMenu = new Menu();
12          handlerObject.fillMenu(fileMenu, xmlSpyMenu.getSubCommands());
13          fileMenu.setLabel(xmlSpyMenu.getLabel().replace("&",""));
14          mb.add(fileMenu);
15          frame.setMenuBar(mb);
16          frame.validate();
17        }catch (AutomationException e1){
18          e1.printStackTrace();
19        }
20        // Disable the button when the action has been performed
21        ((AbstractButton)e.getSource()).setEnabled(false);
22      }
23    });
24  /**
25   * Populates a menu with the commands and submenus contained in an MapForceCommands object
26   */
27  public void fillMenu(Menu newMenu, MapForceCommands mapForceMenu) throws AutomationException
28  {
29    // For each command/submenu in the mapForceMenu
30    for (int i=0;i<mapForceMenu.getCount();++i)
31    {
32      MapForceCommand mapForceCommand = mapForceMenu.getItem(i);
33      if (mapForceCommand.getIsSeparator())
34        newMenu.addSeparator();
35      else
36      {
37        MapForceCommands subCommands = mapForceCommand.getSubCommands();
38        // Is it a command (leaf), or a submenu?
39        if (subCommands.isNull() || subCommands.getCount()==0)
40        {
41          // Command -> add it to the menu, set its ActionCommand to its ID and store it in the menuMap
42          MenuItem mi=new MenuItem(mapForceCommand.getLabel().replace("&",""));
43          mi.setActionCommand(""+mapForceCommand.getID());
44          mi.addActionListener(this);
45          newMenu.add(mi);
46          menuMap.put(mapForceCommand.getID(),mi);
47        }
48        else
49        {
50          // Submenu -> create submenu and repeat recursively
51          Menu newSubMenu = new Menu();
52          fillMenu(newSubMenu, subCommands);
53          newSubMenu.setLabel(mapForceCommand.getLabel().replace("&",""));
54          newMenu.add(newSubMenu);
55        }
56      }
```

```
57    }
58   }
59   /**
60    * Action handler for the menu items
61    * Called when the user selects a menu item; the item's action command corresponds to the command table for
MapForce
62    */
63   public void actionPerformed(ActionEvent e)
64   {
65    try
66    {
67     int iCmd = Integer.parseInt(e.getActionCommand());
68     // Handle explicitly the Close commands
69     switch (iCmd)
70     {
71      case 57602:    // Close
72      case 34050:    // Close All
73       MapForceContainer.initMapForceDocument();
74       break;
75      default:
76       MapForceContainer.mapForceControl.exec(iCmd);
77       break;
78     }
79    }
80    catch (Exception ex)
81    {
82     ex.printStackTrace();
83    }
84
85   }
```

18.5.3.6 UI Update Event Handling

The code listing below shows how a UI-Update event handler can be created.

```
01 /**
02   * Call-back from the MapForceControl.
03   * Called to enable/disable commands
04   */
05  @Override
06  public void onUpdateCmdUI() throws AutomationException
07  {
08   // A command should be enabled if the result of queryStatus contains the Supported (1) and Enabled (2) flags
09   for (java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet())
10    pair.getValue().setEnabled((MapForceContainer.mapForceControl.queryStatus(pair.getKey()) > 2);
11  }
12 /**
13   * Call-back from the MapForceControl.
14   * Usually called while enabling/disabling commands due to UI updates
15   */
16  @Override
17  public boolean onIsActiveEditor(String i_strFilePath) throws AutomationException
18  {
19   try {
20    return MapForceContainer.mapForceDocument.getDocument().getFullName().equalsIgnoreCase(i_strFilePath);
21   }catch (Exception e){
22    return false;
23   }
24  }
```

18.5.3.7 Listing the Properties of a MapForce Mapping

The listing below shows how a Mapping object in MapForce can be loaded as a table and prepared for modal activation.

```
01 //access MapForce Java-COM bridge
02 import com.altova.automation.MapForce.*;
03 import com.altova.automation.MapForce.Component;
```

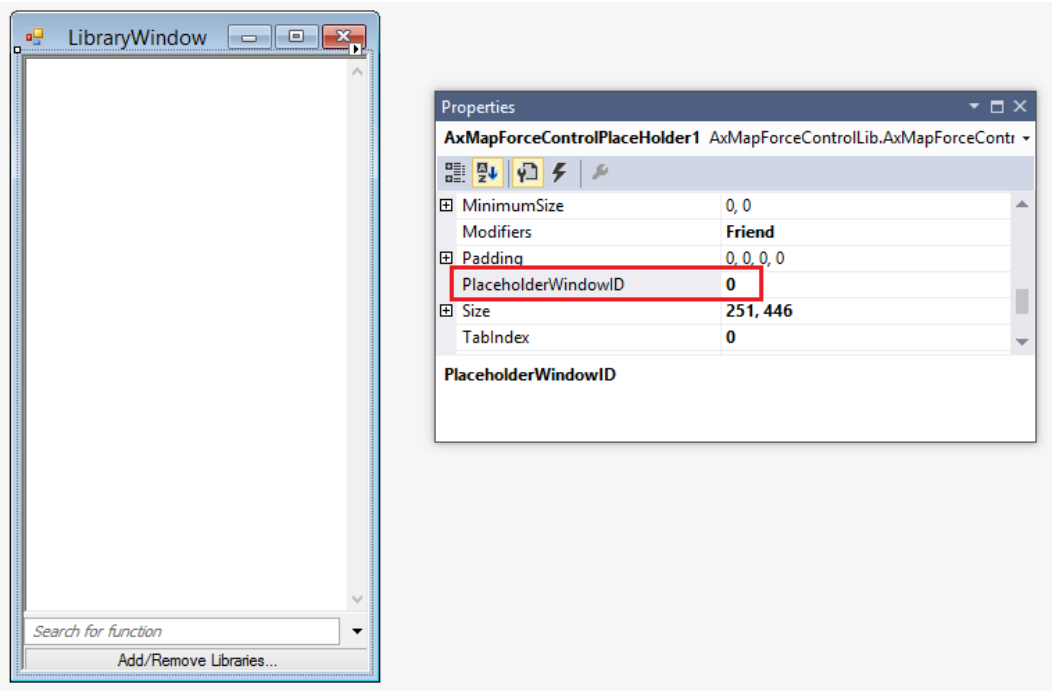
```
04 import com.altova.automation.MapForceEnums.ENUMComponentUsageKind;
05
06 //access AWT and Swing components
07 import java.awt.*;
08 import javax.swing.*;
09 import javax.swing.table.*;
10
11
12 /**
13  * A simple example of a table control loading the structure from a Mapping object.
14  * The class receives an Mapping object, loads its components in a JTable, and prepares
15  * for modal activation.
16  *
17  * Feel free to modify and extend this sample.
18  *
19  * @author Altova GmbH
20  */
21 class MapForceTable extends JDialog
22 {
23  /**
24   * The table control
25   */
26  private JTable myTable;
27
28  /**
29   * Constructor that prepares the modal dialog containing the filled table control
30   * @param mapping The data to be displayed in the table
31   * @param parent Parent frame
32   */
33  public MapForceTable(Mapping mapping, Frame parent)
34  {
35   // Construct the modal dialog
36   super(parent, "MapForce component table", true);
37   // Build up the tree
38   fillTable(mapping);
39   // Arrange controls in the dialog
40   setContentPane(new JScrollPane(myTable));
41  }
42
43  /**
44   * Loads the components of a Mapping object in the table
45   * @param mapping Source data
46   */
47  private void fillTable(Mapping mapping)
48  {
49   try
50   {
51    // count how many Instance components do we have
52    int size = 0;
53    for (Component comp : mapping.getComponents())
54     if (comp.getUsageKind() == ENUMComponentUsageKind.eComponentUsageKind_Instance)
55      ++size;
56
57    // Prepare data
58    final String[] columnNames = {"Component", "Has inputs", "Has outputs", "Input file", "Output file", "Schema"};
59    final Object[][] data = new Object[size][7];
60    int index = 0;
61    for (Component comp : mapping.getComponents())
62     if (comp.getUsageKind() == ENUMComponentUsageKind.eComponentUsageKind_Instance)
63     {
64      int i=0;
65      data[index][i++] = comp.getName();
66      data[index][i++] = new Boolean(comp.getHasIncomingConnections());
67      data[index][i++] = new Boolean(comp.getHasOutgoingConnections());
68      data[index][i++] = comp.getInputInstanceFile();
69      data[index][i++] = comp.getOutputInstanceFile();
70      data[index++][i] = comp.getSchema();
71     }
72
73    // Set up table
74    myTable = new JTable(new AbstractTableModel(){
75       public String getColumnName(int col){return columnNames[col];}
76       public int getRowCount(){return data.length;}
77       public int getColumnCount(){return columnNames.length;}
78       public Object getValueAt(int row, int col){return data[row][col];}
79       public boolean isCellEditable(int row, int col){return false;}
80       public Class getColumnClass(int c){return getValueAt(0,c).getClass();}
81    });
82
83    // Set width
84    for(index = 0; index < columnNames.length; ++index)
85     myTable.getColumnModel().getColumn(index).setMinWidth(80);
```

```
86      myTable.getColumnModel().getColumn(5).setMinWidth(400);
87    }
88    catch (Exception e)
89    {
90      e.printStackTrace();
91    }
92  }
93
94 }
```

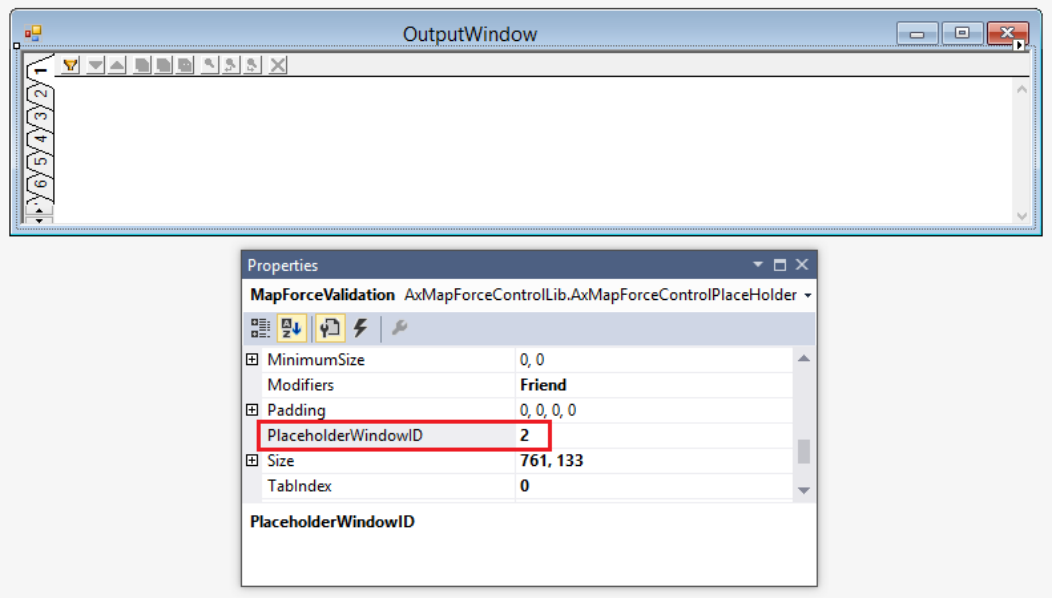
18.5.4 VB.NET

Source code which illustrates integration of MapForceControl into a VB.NET application can be found in the folder `<ApplicationFolder>\Examples\ActiveX\VB.NET` of your MapForce installation. The solution consists of three windows, as follows:

1. **MainWindow.vb** - the main document window, which also includes a basic application menu.
2. **LibraryWindow.vb** - the Library window. The contents of this window is populated by a Placeholder control which has the **PlaceholderWindowID** property set to 0 (this value instructs the control to display specifically the Library window).



3. **OutputWindow.vb** - the Messages (Output) window. The contents of this window is populated by a Placeholder control which has the **PlaceholderWindowID** property set to 2 (this value instructs the control to display specifically the Output window).



Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

For the list of prerequisites, see Prerequisites.

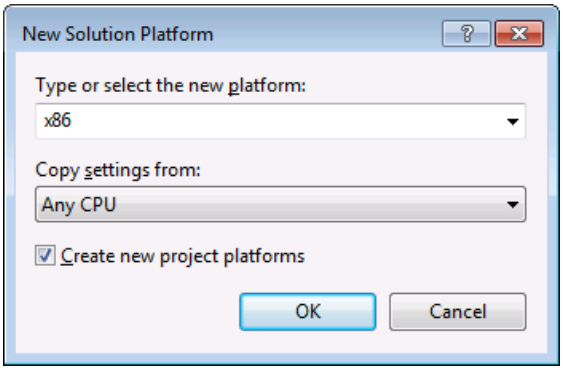
Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

18.6 Command Reference

This section lists the names and identifiers of all menu commands that are available within MapForce. Every sub-section lists the commands from the corresponding top-level menu of MapForce. The command tables are organized as follows:

- The "Menu Item" column shows the command's menu text as it appears in MapForce, to make it easier for you to identify the functionality behind the command.
- The "Command Name" column specifies the string that can be used to get an icon with the same name from **ActiveX\Images** folder of the MapForce installation directory.
- The "ID" column shows the numeric identifier of the column that must be supplied as argument to methods which execute or query this command.

To execute a command, use the `MapForceControl.Exec` or the `MapForceControlDocument.Exec` methods. To query the status of a command, use the `MapForceControl.QueryStatus` or `MapForceControlDocument.QueryStatus` methods.

Depending on the edition of MapForce you have installed, some of these commands might not be supported.

18.6.1 "File" Menu

The "File" menu has the following commands:

Menu item	Command name	ID
New...	ID_FILE_NEW	57600
Open...	ID_FILE_OPEN	57601
Save	ID_FILE_SAVE	57603
Save As...	ID_FILE_SAVE_AS	57604
Save All	ID_FILE_SAVEALL	32377
Reload	IDC_FILE_RELOAD	32467
Close	ID_WINDOW_CLOSE	32453
Close All	ID_WINDOW_CLOSEALL	32454
Print...	ID_FILE_PRINT	57607
Print Preview	ID_FILE_PRINT_PREVIEW	57609
Print Setup...	ID_FILE_PRINT_SETUP	57606
Validate Mapping	ID_MAPPING_VALIDATE	32347
Mapping Settings	ID_MAPPING_SETTINGS	32396
Generate Code in Selected Language	ID_FILE_GENERATE_SELECTED_CODE	32362
XSLT 1.0	ID_FILE_GENERATEXSLT	32360

Menu item	Command name	ID
XSLT 2.0	ID_FILE_GENERATEXSLT2	32361
XQuery	ID_FILE_GENERATEXQUERY	32359
Java	ID_FILE_GENERATEJAVACODE	32358
C# (Sharp)	ID_FILE_GENERATECSCODE	32357
C++	ID_FILE_GENERATECPPCODE	32356
Compile to MapForce Server Execution File...	ID_FILE_CREATE_SERVER_EXECUTION_FILE	32517
Deploy to FlowForce Server...	ID_FILE_DEPLOY_MAPPING	32506
Generate Documentation...	ID_FILE_GENERATE_DOCUMENTATION	32468
Recent File	ID_FILE_MRU_FILE1	57616
Exit	ID_APP_EXIT	57665

18.6.2 "Edit" Menu

The "Edit" menu has the following commands:

Menu item	Command name	ID
Undo	ID_EDIT_UNDO	57643
Redo	ID_EDIT_REDO	57644
Find...	ID_EDIT_FIND	57636
Find Next	ID_EDIT_FINDNEXT	32349
Find Previous	ID_EDIT_FINDPREV	32350
Cut	ID_EDIT_CUT	57635
Copy	ID_EDIT_COPY	57634
Paste	ID_EDIT_PASTE	57637
Delete	ID_EDIT_CLEAR	57632
Select All	ID_EDIT_SELECT_ALL	57642

18.6.3 "Insert" Menu

The "Insert" menu has the following commands:

Menu item	Command name	ID
XML Schema/File...	ID_INSERT_XSD	32393

Menu item	Command name	ID
Database...	ID_INSERT_DATABASE	32389
EDI...	ID_INSERT_EDI	32390
Text File...	ID_INSERT_TXT	32392
Web Service Function...	ID_INSERT_WEBSERVICE_FUNCTION	32319
Excel 2007+ File...	ID_INSERT_EXCEL	32376
XBRL Document...	ID_INSERT_XBRL	32469
JSON Schema/File...	ID_INSERT_JSON	32531
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402
Constant...	ID_INSERT_CONSTANT	32388
Variable...	ID_INSERT_VARIABLE	32500
Join	ID_INSERT_JOIN	32581
Sort: Nodes/Rows	ID_INSERT_SORT	32444
Filter: Nodes/Rows	ID_INSERT_FILTER	32391
SQL-WHERE/ORDER	ID_INSERT_SQLWHERE_CONDITION	32351
Value-Map	ID_INSERT_VALUEMAP	32354
IF-Else Condition	ID_INSRT_CONDITION	32394
Exception	ID_INSERT_EXCEPTION	32311

18.6.4 "Project" Menu

The "Project" menu has the following commands:

Menu item	Command name	ID
Reload Project	ID_PROJECT_RELOAD	32476
Close Project	ID_FILE_CLOSEPROJECT	32355
Save Project	ID_FILE_SAVEPROJECT	32378
Add Files to Project...	ID_PROJECT_ADDFILESTOPROJECT	32420
Add Active File to Project	ID_PROJECT_ADDACTIVEFILETOPROJECT	32419
Create Folder...	ID_PROJECT_CREATE_FOLDER	32310
Open Mapping	ID_PROJECT_OPEN_MAPPING	32307
Create Mapping for Operation...	ID_PROJECT_CREATE_MAPPING_FOR_OPERATION	32399
Add Mapping File for Operation...	ID_PROJECT_ADD_MAPPING	32309

Menu item	Command name	ID
Insert Web Service...	ID_PROJECT_INSERT_WEBSERVICE	32306
Open File in XMLSpy	ID_PROJECT_OPEN_IN_XMLSPY	32305
Generate Code for Entire Project	ID_PROJECT_GENERATE_ALL	32303
XSLT 1.0	ID_PROJECT_GENERATEXSLTCODE_ENTIRE	32408
XSLT 2.0	ID_PROJECT_GENERATEXSLT2CODE_ENTIRE	32409
XQuery	ID_PROJECT_GENERATEXQUERYCODE_ENTIRE	32410
Java	ID_PROJECT_GENERATEJAVACODE_ENTIRE	32411
C# (Sharp)	ID_PROJECT_GENERATECSCODE_ENTIRE	32412
C++	ID_PROJECT_GENERATECPPCODE_ENTIRE	32413
Properties	ID_PROJECT_PROPERTIES	32404
Recent Project	ID_FILE_MRU_PROJECT1	32364

18.6.5 "Component" Menu

The "Component" menu has the following commands:

Menu item	Command name	ID
Change Root Element...	ID_COMPONENT_CHANGEROOTELEMENT	32334
Edit Schema Definition in XMLSpy	ID_COMPONENT_EDIT_SCHEMA	32337
Edit FlexText Configuration	ID_COMPONENT_EDIT_MFT	32301
Add/Remove/Edit Database Objects...	ID_COMPONENT_SELECTTABLES	32346
Create Mapping to EDI X12 997	ID_COMPONENT_CREATE_MAPPING_TO_997	32483
Create Mapping to EDI X12 999	ID_COMPONENT_CREATE_MAPPING_TO_999	32484
Refresh	IDC_COMMAND_REFRESH_COMPONENT	32373
Add Duplicate Input Before	ID_COMPONENT_CREATE_DUPLICATE_ICON_BEFORE	32503
Add Duplicate Input After	ID_COMPONENT_CREATE_DUPLICATE_ICON	32335
Remove Duplicate	ID_COMPONENT_REMOVE_DUPLICAT	32339

Menu item	Command name	ID
	E_ICON	
Add Comment Before	ID_COMPONENT_ADD_COMMENT_BEFORE	32518
Add Comment After	ID_COMPONENT_ADD_COMMENT_AFTER	32519
Add Processing Instruction Before...	ID_COMPONENT_ADD_PI_BEFORE	32520
Add Processing Instruction After...	ID_COMPONENT_ADD_PI_AFTER	32521
Edit Processing Instruction Name...	ID_COMPONENT_EDIT_PI	32524
Delete Comment/Processing Instruction	ID_COMPONENT_REMOVE_COMMENT_PI	32522
Write Content as CDATA Section	ID_COMPONENT_TOGGLE_CDATA	32525
Database Table Actions	ID_POPUP_DATABASETABLEACTIONS	32400
Query Database...	ID_QUERY_DATABASE	32341
Align Tree Left	ID_COMPONENT_LEFTALIGNTREE	32338
Align Tree Right	ID_COMPONENT_RIGHTALIGNTREE	32340
Properties	ID_COMPONENT_PROPERTIES	32336

18.6.6 "Connection" Menu

The "Connection" menu has the following commands:

Menu item	Command name	ID
Auto Connect Matching Children	ID_CONNECTION_AUTOCONNECTCHILDREN	32342
Settings for Connect Matching Children	ID_CONNECTION_SETTINGS	32344
Connect Matching Children...	ID_CONNECTION_MAPCHILDELEMENTS	32343
Target Driven (Standard)	ID_POPUP_NORMALCONNECTION	32401
Copy-All (Copy Child Items)	ID_POPUP_NORMALWITHCHILDREN_CONNECTION	32460
Source Driven (Mixed Content)	ID_POPUP_ORDERBYSOURCECONNECTION	32403
Properties	ID_POPUP_CONNECTION_SETTINGS	32398

18.6.7 "Function" Menu

The "Function" menu has the following commands:

Menu item	Command name	ID
Create User-Defined Function...	ID_FUNCTION_CREATE_EMPTY	32380
Create User-Defined Function from Selection...	ID_FUNCTION_CREATE_FROM_SELECTION	32381
Function Settings	ID_FUNCTION_SETTINGS	32387
Remove Function	ID_FUNCTION_REMOVE	32385
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402

18.6.8 "Output" Menu

The "Output" menu has the following commands:

Menu item	Command name	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
Built-In Execution Engine	ID_SELECT_LANGUAGE_BUILTIN	32490
Validate Output File	ID_XML_VALIDATE	32458
Save Output File...	IDC_FILE_SAVEGENERATEDOUTPUT	32321
Save All Output Files...	IDC_FILE_SAVEALLGENERATEDOUTPUT	32374
Regenerate Output	ID_REGENERATE_PREVIEW_OUTPUT	32480
Run SQL-Script	ID_TRANSFORM_RUN_SQL	32442
Insert/Remove Bookmark	ID_TOGGLE_BOOKMARK	32317
Next Bookmark	ID_GOTONEXTBOOKMARK	32315
Previous Bookmark	ID_GOTOPREVBOOKMARK	32314
Remove All Bookmarks	ID_REMOVEALLBOOKMARKS	32313
Pretty-Print XML Text	ID_PRETTY_PRINT_OUTPUT	32363

Menu item	Command name	ID
Text View Settings	ID_TEXTVIEWSETTINGSDIALOG	32472

18.6.9 "Debug" Menu

The "Debug" menu has the following commands:

Menu item	Command name	ID
Start Debugging	ID_DEBUG_START	32540
Stop Debugging	ID_DEBUG_STOP	32541
Step Into	ID_DEBUG_STEP_INTO	32545
Step Over	ID_DEBUG_STEP_OVER	32551
Step Out	ID_DEBUG_STEP_OUT	32552
Minimal Step	ID_DEBUG_STEP_NEXT_TRACE	32554

18.6.10 "View" Menu

The "View" menu has the following commands:

Menu item	Command name	ID
Show Annotations	ID_SHOW_ANNOTATION	32435
Show Types	ID_SHOW_TYPES	32437
Show Library in Function Header	ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER	32448
Show Tips	ID_SHOW_TIPS	32436
XBRL Display Options	ID_VIEW_XBRL_DISPLAY_OPTIONS	32473
Show Selected Component Connectors	ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS	32443
Show Connectors from Source to Target	ID_VIEW_RECURSIVEAUTOHIGHLIGHT	32447
Zoom...	ID_VIEW_ZOOM	32451
Back	ID_CMD_BACK	32479
Forward	ID_CMD_FORWARD	32478
Status Bar	ID_VIEW_STATUS_BAR	59393
Library Window	ID_VIEW_LIBRARY_WINDOW	32445
Messages	ID_VIEW_VALIDATION_OUTPUT	32450
Overview	ID_VIEW_OVERVIEW_WINDOW	32446

Menu item	Command name	ID
Project Window	ID_VIEW_PROJECT_WINDOW	32302
Values	ID_DEBUG_VIEW_VALUES_WINDOW	32544
Context	ID_DEBUG_VIEW_CONTEXT_WINDOW	32546
Breakpoints	ID_DEBUG_VIEW_DEBUGPOINTS_WINDOW	32547

18.6.11 "Tools" Menu

The "Tools" menu has the following commands:

Menu item	Command name	ID
Global Resources	IDC_GLOBALRESOURCES	37401
	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Create Reversed Mapping	ID_CREATE_REVERSED_MAPPING	32489
Customize...	IDC_APP_TOOLS_CUSTOMIZE	32959
Options...	ID_TOOLS_OPTIONS	32441

18.6.12 "Window" Menu

The "Window" menu has the following commands:

Menu item	Command name	ID
Cascade	ID_WINDOW_CASCADE	57650
Tile Horizontal	ID_WINDOW_TILE_HORZ	57651
Tile Vertical	ID_WINDOW_TILE_VERT	57652

18.6.13 "Help" Menu

The "Help" menu has the following commands:

Menu item	Command name	ID
Table of Contents...	IDC_HELP_CONTENTS	32322
Index...	IDC_HELP_INDEX	32323
Search...	IDC_HELP_SEARCH	32324
Software Activation...	IDC_ACTIVATION	32701

Menu item	Command name	ID
Order Form...	IDC_OPEN_ORDER_PAGE	32326
Registration...	IDC_REGISTRATION	32330
Check for Updates...	IDC_CHECK_FOR_UPDATES	32700
Support Center...	IDC_OPEN_SUPPORT_PAGE	32327
FAQ on the Web...	IDC_SHOW_FAQ	32331
Download Components and Free Tools...	IDC_OPEN_COMPONENTS_PAGE	32325
MapForce on the Internet..	IDC_OPEN_XML_SPY_HOME	32328
MapForce Training...	IDC_OPEN_MAPFORCE_TRAINING_PAGE	32300
About MapForce...	ID_APP_ABOUT	57664

18.7 Object Reference

Objects:
MapForceCommand
MapForceCommands
MapForceControl
MapForceControlDocument
MapForceControlPlaceHolder

To give access to standard MapForce functionality, objects of the **MapForce automation interface** can be accessed as well. See MapForceControl.Application, MapForceControlDocument.Document and MapForceControlPlaceHolder.Project for more information.

18.7.1 MapForceCommand

Properties:
ID
Label
Name
IsSeparator
ToolTip
StatusText
Accelerator
SubCommands

Description:
A command object can be one of the following: an executable command, a command container (for example, a menu, submenu, or toolbar), or a menu separator. To determine what kind of information is stored in the current `Command` object, query its `ID`, `IsSeparator`, and `SubCommands` properties, as follows.

The Command object is...	When...
An executable command	• `ID` is greater than zero • `IsSeparator` is false • `SubCommands` is empty
A command container	• `ID` is zero • `IsSeparator` is true • `SubCommands` contains a collection of `Command` objects.
Separator	• `ID` is zero • `IsSeparator` is true

18.7.1.1 Accelerator

Property: Accelerator as string

Description:
Returns the accelerator key defined for the command. If the command has no accelerator key assigned, this property returns the empty string. The string representation of the accelerator key has the following format:

```
[ALT+][CTRL+][SHIFT+]key
```

Where `key` is converted using the Windows Platform SDK function `GetKeyNameText`.

18.7.1.2 ID

Property: `ID` as long

Description:
This property gets the unique identifier of the command. A command's ID is required to execute the command (using `Exec`) or query its status (using `QueryStatus`). If the command is a container for other commands (for example, a top-level menu), or a separator, the ID is 0.

18.7.1.3 IsSeparator

Property: `IsSeparator` as boolean

Description:
The property returns `true` if the command object is a menu separator; `false` otherwise. See also Command.

18.7.1.4 Label

Property: `Label` as string

Description:
This property gets the text of the command as it is displayed in the graphical user interface of MapForce. If the command is a separator, "Label" is an empty string. This property may also return an empty string for some toolbar commands that do not have any GUI text associated with them.

18.7.1.5 Name

Property: `Name` as string

Description:
This property gets the unique name of the command. This value can be used to get the icon file of the command, where it is available. The available icon files can be found in the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your MapForce installation.

18.7.1.6 StatusText

Property: `Label` as string

Description:
The status text is the text shown in the status bar of MapForce when the command is selected. It applies only to command objects that are not separators or containers of other commands; otherwise, the property is an empty string.

18.7.1.7 SubCommands

Property: SubCommands as Commands

Description:
The `SubCommands` property gets the collection of `Command` objects that are sub-commands of the current command. The property is applicable only to commands that are containers for other commands (menus, submenus, or toolbars). Such container commands have the `ID` set to 0, and the `IsSeparator` property set to `false`.

18.7.1.8 ToolTip

Property: ToolTip as string

Description:
This property gets the text that is shown as a tool-tip for each command. If the command does not have a tooltip text, the property returns an empty string.

18.7.2 MapForceCommands

Properties:
Count
Item

Description:
Collection of Command objects to get access to command labels and IDs of the MapForceControl. Those commands can be executed with the Exec method and their status can be queried with QueryStatus.

18.7.2.1 Count

Property: Count as long

Description:
Number of Command objects on this level of the collection.

18.7.2.2 Item

Property: Item (n as long) as Command

Description:
Gets the command with the index n in this collection. Index is 1-based.

18.7.3 MapForceControl

Properties:
IntegrationLevel
Appearance
Application
BorderStyle
CommandsList
CommandsStructure (deprecated)
EnableUserPrompts
MainMenu
Toolbars

Methods:
Open
Exec
QueryStatus

Events:
OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged
OnDocumentOpened
OnValidationWindowUpdated

This object is a complete ActiveX control and should only be visible if the MapForce library is used in the Application Level mode.

CLSID:A38637E9-5759-4456-A167-F01160CC22C1
ProgID:AltovaMapForceControl

18.7.3.1 Properties

The following properties are defined:

IntegrationLevel
EnableUserPrompts
Appearance
BorderStyle

Command related properties:
CommandsList
MainMenu
Toolbars
CommandsStructure (deprecated)

Access to MapForceAPI:
Application

Appearance

Property: Appearance as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the control. Default value is `0`.

Application

Property: Application as Application

Dispatch Id: *1*

Description:
The Application property gives access to the Application object of the complete MapForce automation server API. The property is read-only.

BorderStyle

Property: BorderStyle as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

CommandsList

Property: CommandList as `Commands` (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with MapForceControl. To get commands organized according to their menu structure, use `MainMenu`. To get toolbar commands, use `Toolbars`.

```
public void GetAllMapForceCommands()
{
    // Get all commands from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommands commands =
this.axMapForceControl1.CommandList;
    // Iterate through all commands
```

```
    for (int i = 0; i < commands.Count; i++)
    {
      // Get each command by index and output it to the console
      MapForceControlLib.MapForceCommand cmd =
axMapForceControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name,
cmd.Label.Replace("&", ""));
    }
}
```

C# example

EnableUserPrompts

Property: EnableUserPrompts as boolean

Dispatch Id: *1006*

Description:
Setting this property to *false*, disables user prompts in the control. The default value is *true*.

IntegrationLevel

Property: IntegrationLevel as ICActiveXIntegrationLevel

Dispatch Id: *1000*

Description:
The IntegrationLevel property determines the operation mode of the control. See also Integration at Application Level and Integration at Document Level for more information.

Note: It is important to set this property immediately after the creation of the MapForceControl object.

MainMenu

Property: MainMenu as Command (read-only)

Dispatch Id: *1003*

Description:
This property provides information about the structure and commands available in the MapForceControl main menu, as a Command object. The Command object contains all available submenus of MapForce (for example "File", "Edit", "View" etc.). To access the submenu objects, use the SubCommands property of the MainMenu property. Each submenu is also a Command object. For each submenu, you can then further iterate through their SubCommands property in order to get their corresponding child commands and separators (this technique may be used, for example, to create the application menu programmatically). Note that some menu commands act as containers ("parents") for other menu commands, in which case they also have a SubCommands property. To get the structure of all menu commands programmatically, you will

likely need to create a recursive function.

```
public void GetMapForceMenus()
{
    // Get the main menu from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommand mainMenu =
this.axMapForceControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      MapForceControlLib.MapForceCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including
separators)", menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# example

Toolbars

Property: Toolbars as Commands (read-only)

Dispatch Id: *1005*

Description:
This property provides information about the structure of MapForceControl toolbars, as a `Command` object. The `Command` object contains all available toolbars of MapForce. To access the toolbars, use the `SubCommands` property of the `Toolbars` property. Each toolbar is also a `Command` object. For each toolbar, you can then further iterate through their `SubCommands` property in order to get their commands (this technique may be used, for example, to create the application's toolbars programmatically).

```
public void GetMapForceToolbars()
{
    // Get the application toolbars from the MapForce ActiveX control
assigned to the current form
    MapForceControlLib.MapForceCommands toolbars =
this.axMapForceControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      MapForceControlLib.MapForceCommand toolbar = toolbars[i];
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         MapForceControlLib.MapForceCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if (!cmd.IsSeparator)
         {
```

```
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name,
cmd.Label.Replace("&", ""));
        }
      }
    }
}
```

C# example

18.7.3.2 Methods

The following methods are defined:

Open
Exec
QueryStatus

Exec

Method: Exec (nCmdID as long) as boolean

Dispatch Id: *6*

Description:
This method calls the MapForce command with the ID `nCmdID`. If the command can be executed, the method returns `true`. To get a list of all available commands, use `CommandsList`. To retrieve the status of any command, use `QueryStatus`.

Open

Method: Open (strFilePath as string) as boolean

Dispatch Id: *5*

Description:
The result of the method depends on the extension passed in the argument strFilePath. If the file extension is .sps, a new document is opened. If the file extension is .svp, the corresponding project is opened. If a different file extension is passed into the method, the control tries to load the file as a new component into the active document.

Do not use this method to load documents or projects when using the control in document-level integration mode. Instead, use MapForceControlDocumentOpen and MapForceControlPlaceHolderOpenProject.

QueryStatus

Method: QueryStatus (nCmdID as long) as long

Dispatch Id: *7*

Description:

QueryStatus returns the enabled/disabled and checked/unchecked status of the command specified by nCmdID. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if QueryStatus returns 0 the command ID is not recognized as a valid MapForce command. If QueryStatus returns a value of 1 or 5, the command is disabled.

18.7.3.3 Events

The MapForceControl ActiveX control provides the following connection point events:

OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged

OnDocumentOpened
OnValidationWindowUpdated

OnCloseEditingWindow

Event: OnCloseEditingWindow (i_strFilePath as String) as boolean

Dispatch Id: *1002*

Description:
This event is triggered when MapForce needs to close an already open document. As an answer to this event, clients should close the editor window associated with *i_strFilePath*. Returning *true* from this event indicates that the client has closed the document. Clients can return *false* if no specific handling is required and MapForceControl should try to close the editor and destroy the associated document control.

OnContextChanged

Event: OnContextChanged (i_strContextName as String, ibActive as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in MapForce

OnDocumentOpened

Event: `OnDocumentOpened (objDocument as Document)`

Dispatch Id: *1*

Description:
This event is triggered whenever a document is opened. The argument `objDocument` is a `Document` object from the MapForce automation interface and can be used to query for more details about the document, or perform additional operations. When integrating on document-level, it is often better to use the event `MapForceControlDocument.OnDocumentOpened` instead.

OnFileChangedAlert

Event: `OnFileChangedAlert(i_strFilePath as String) as bool`

Dispatch Id: *1001*

Description:
This event is triggered when a file loaded with MapForceControl is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

OnLicenseProblem

Event: `OnLicenseProblem (i_strLicenseProblemText as String)`

Dispatch Id: *1005*

Description:
This event is triggered when MapForceControl detects that no valid license is available for this control. In case of restricted user licenses this can happen some time after the control has been initialized. Integrators should use this event to disable access to this control's functionality. After returning from this event, the control will block access to its functionality (e.g. show empty windows in its controls and return errors on requests).

OnOpenedOrFocused

Event: `OnOpenedOrFocused (i_strFilePath as String, i_bOpenWithThisControl as bool)`

Dispatch Id: *1000*

Description:
When integrating at application level, this event informs clients that a document has been opened, or made active by MapForce.

When integrating at document level, this event instructs the client to open the file `i_strFilePath` in a document window. If the file is already open, the corresponding document window should be made the active window.

if `i_bOpenWithThisControl` is `true`, the document must be opened with MapForceControl, since internal access is required. Otherwise, the file can be opened with different editors.

OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd as long )`

Dispatch Id: *1006*

Description:
This event is triggered when the tool window is updated.

OnUpdateCmdUI

Event: `OnUpdateCmdUI()`

Dispatch Id: *1003*

Description:
Called frequently to give integrators a good opportunity to check status of MapForce commands using `MapForceControlQueryStatus`. Do not perform long operations in this callback.

OnValidationWindowUpdated

Event: `OnValidationWindowUpdated ()`

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window is updated with new information.

18.7.4 MapForceControlDocument

Properties:
Appearance
BorderStyle
Document
IsModified
Path
ReadOnly

Methods:
Exec
New
Open
QueryStatus
Reload
Save

SaveAs

Events:
OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate

If the MapForceControl is integrated in the Document Level mode each document is displayed in an own object of type `MapForceControlDocument`. The `MapForceControlDocument` contains only one document at the time but can be reused to display different files one after another.

This object is a complete ActiveX control.

`CLSID:DFBB0871-DAFE-4502-BB66-08CEB7DF5255`
`ProgID:AltovaMapForceControlDocument`

18.7.4.1 Properties

The following properties are defined:

ReadOnly
IsModified
Path
Appearance
BorderStyle

Access to MapForceAPI:
Document

Appearance

Property: Appearance as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the document control. Default value is `0`.

BorderStyle

Property: BorderStyle as short

Dispatch Id: *-504*

Description:
A value of 1 displays the control with a thin border. Default value is `0`.

Document

Property: Document as Document

Dispatch Id: *1*

Description:
The Document property gives access to the Document object of the MapForce automation server API. This interface provides additional functionality which can be used with the document loaded in the control. The property is read-only.

IsModified

Property: IsModified as boolean (read-only)

Dispatch Id: *1006*

Description:
IsModified is *true* if the document content has changed since the last open, reload or save operation. It is *false*, otherwise.

Path

Property: Path as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

ReadOnly

Property: ReadOnly as boolean

Dispatch Id: *1007*

Description:
Using this property you can turn on and off the read-only mode of the document. If ReadOnly is true it is not possible to do any modifications.

18.7.4.2 Methods

The following methods are defined:

Document handling:

New
Open
Reload
Save
SaveAs

Command Handling:
Exec
QueryStatus

Exec

Method: Exec (nCmdID as long) as boolean

Dispatch Id: *8*

Description:
Exec calls the MapForce command with the ID nCmdID. If the command can be executed, the method returns true. This method should be called only if there is currently an active document available in the application.

To get commands organized according to their menu structure, use the MainMenu property of MapForceControl. To get toolbar commands, use the Toolbars property of the MapForceControl.

New

Method: New () as boolean

Dispatch Id: *1000*

Description:
This method initializes a new document inside the control..

Open

Method: Open (strFileName as string) as boolean

Dispatch Id: *1001*

Description:
Open loads the file strFileName as the new document into the control.

QueryStatus

Method: QueryStatus (nCmdID as long) as long

Dispatch Id: *9*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid MapForce command. If `QueryStatus` returns a value of `1` or `5` the command is disabled. The client should call the `QueryStatus` method of the document control if there is currently an active document available in the application.

Reload

Method: `Reload` () as boolean

Dispatch Id: *1002*

Description:
`Reload` updates the document content from the file system.

Save

Method: Save () as boolean

Dispatch Id: *1003*

Description:
Save saves the current document at the location Path.

SaveAs

Method: `SaveAs` (`strFileName` as string) as boolean

Dispatch Id: *1004*

Description:
SaveAs sets Path to *strFileName* and then saves the document to this location.

18.7.4.3 Events

The MapForceControlDocument ActiveX control provides following connection point events:

OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate
OnSetEditorTitle

OnActivate

Event: OnActivate ()

Dispatch Id: *1005*

Description:
This event is triggered when the document control is activated, has the focus, and is ready for user input.

OnContextChanged

Event: OnContextChanged (i_strContextName as String, ib_Active as bool) as bool

Dispatch Id: *1004*

Description: None

OnDocumentClosed

Event: OnDocumentClosed (objDocument as Document)

Dispatch Id: *1001*

Description:
This event is triggered whenever the document loaded into this control is closed. The argument objDocument is a Document object from the MapForce automation interface and should be used with care.

OnDocumentOpened

Event: OnDocumentOpened (objDocument as Document)

Dispatch Id: *1000*

Description:
This event is triggered whenever a document is opened in this control. The argument

objDocument is a Document object from the MapForce automation interface, and can be used to query for more details about the document, or perform additional operations.

OnDocumentSaveAs

Event: OnContextDocumentSaveAs(i_strFileName as String)

Dispatch Id: *1007*

Description:
This event is triggered when this document gets internally saved under a new name.

OnFileChangedAlert

Event: OnFileChangedAlert() as bool

Dispatch Id: *1003*

Description:
This event is triggered when the file loaded into this document control is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

OnModifiedFlagChanged

Event: OnModifiedFlagChanged (ibIsModified as boolean)

Dispatch Id: *1002*

Description:
This event gets triggered whenever the document changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the document contents differs from the original content, and *false*, otherwise.

OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

18.7.5 MapForceControlPlaceHolder

Properties available for all kinds of placeholder windows:
PlaceholderWindowID

Properties for project placeholder window:
Project

Methods for project placeholder window:
OpenProject
CloseProject

The MapForceControlPlaceHolder control is used to show the additional MapForce windows like Overview, Library or Project window. It is used like any other ActiveX control and can be placed anywhere in the client application.

CLSID:FDEC3B04-05F2-427d-988C-F03A85DE53C2
ProgID:AltovaMapForceControlPlaceHolder

18.7.5.1 Properties

The following properties are defined:

PlaceholderWindowID

Access to MapForceAPI:
Project

Label

Property: Label as String (read-only)

Dispatch Id: *1001*

Description:
This property gives access to the title of the placeholder. The property is read-only.

PlaceholderWindowID

Property: PlaceholderWindowID as MapForceControlPlaceholderWindow

Dispatch Id: *1*

Description:
Using this property the object knows which MapForce window should be displayed in the client area of the control. The PlaceholderWindowID can be set at any time to any valid value of the MapForceControlPlaceholderWindow enumeration. The control changes its state immediately and shows the new MapForce window.

Project

Property: Project as Project (read-only)

Dispatch Id: *2*

Description:
The Project property gives access to the Project object of the MapForce automation server API. This interface provides additional functionality which can be used with the project loaded into the control. The property will return a valid project interface only if the placeholder window has PlaceholderWindowID with a value of MapForceXProjectWindow (=3) The property is read-only.

18.7.5.2 Methods

The following method is defined:

OpenProject
CloseProject

OpenProject

Method: OpenProject(strFileName as string) as boolean

Dispatch Id: *3*

Description:
OpenProject loads the file strFileName as the new project into the control. The method will fail if the placeholder window has a PlaceholderWindowID different to XMLSpyXProjectWindow (=3)

CloseProject

Method: CloseProject()

Dispatch Id: *4*

Description:
CloseProject closes the project loaded the control. The method will fail if the placeholder window has a PlaceholderWindowID different to MapForceXProjectWindow (=3)

18.7.5.3 Events

The MapForceControlPlaceholder ActiveX control provides following connection point events:

OnModifiedFlagChanged

OnModifiedFlagChanged

Event: OnModifiedFlagChanged(ibIsModified as boolean)

Dispatch Id: *1*

Description:
This event gets triggered only for placeholder controls with a PlaceholderWindowID of MapForceXProjectWindow (=3). The event is fired whenever the project content changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the project contents differs from the original content, and *false*, otherwise.

OnSetLabel

Event: OnSetLabel(istrNewLabel as string)

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

18.7.6 Enumerations

The following enumerations are defined:

ICActiveXIntegrationLevel
MapForceControlPlaceholderWindow

18.7.6.1 ICActiveXIntegrationLevel

Possible values for the IntegrationLevel property of the MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

18.7.6.2 MapForceControlPlaceholderWindow

This enumeration contains the list of the supported additional MapForce windows.

```
MapForceXNoWindow                = -1
MapForceXLibraryWindow           = 0
MapForceXOverviewWindow          = 1
MapForceXValidationWindow        = 2
MapForceXProjectWindow           = 3
MapForceXDebuggerValuesWindow    = 4
MapForceXDebuggerContextWindow   = 5
```

```
MapForceXDebuggerPointsWindow    = 6
```

Chapter 19

付録

19 付録

以下の付録にはMapForce に関する技術的な情報や、ライセンスに関する重要な情報が収められています。各付録には以下のようにサブセクションが収められています：

技術データ

- OS ならびにメモリの要件
- Altova XML パーサー
- Altova XSLT とXQuery エンジン
- Unicode のサポート
- インターネットの使用
- ライセンス使用状況測定

ライセンス情報

- 電子的なソフトウェアの配布
- 著作権
- 使用許諾契約書

(C) 2017 Altova GmbH

19.1 エンジン情報

このセクションには Altova XML バリデーター、Altova XSLT 1.0 エンジン、Altova XSLT 2.0 エンジン、そして Altova XQuery エンジンの実装に特化した情報が含まれます。

19.1.1 XSLT および XQuery エンジンに関する情報

MapForce の XSLT および XQuery エンジンは、W3C 仕様に従っています。ですから、XMLSpy の以前のバージョン内の Altova エンジンよりも厳密です。この結果、以前のエンジンで無視されていた小さなエラーが、MapForce によりエラーとして挙げられます。

例えば:

- パス演算子の結果がノードと非 -ノードを両方含む場合、型エラー (`err:XPTY0018`) です。
- パス式 `E1/E2` 内の `E1` がノードのシーケンスを評価しない場合、型エラー (`err:XPTY0019`) です。

この種類のエラーが発生した場合、XSLT/XQuery ドキュメントまたはインスタンスドキュメントを必要に応じて修正してください。

このセクションは、エンジンの実装固有の機能を仕様別に整理して説明します。

- XSLT 1.0
- XSLT 2.0
- XQuery 1.0

19.1.1.1 XSLT 1.0

MapForce の XSLT 1.0 エンジンは、World Wide Web Consortium (ワールド・ワイド・ウェブ・コンソーシアム) (W3C) の1999 年 11月 16日版の XSLT 1.0 勧告 および1999 年 11月 16日版の XPath 1.0 勧告 に準拠します。実装に関しての以下の情報に注意してください。

実装についての注意点

`xsl:output` の `method` 属性が HTML に設定された場合、または、が HTML 出力 デフォルトで選択されている場合、内の特殊文字は HTML ドキュメントに HTML 文字参照として出力内に挿入されます。例えば、文字 U+00A0 (ブレーク無しのスペースのための 16進数レファレンス) が HTML コード内に文字の参照 (` ` or ` `) 、または、エンティティ参照 ` ` として挿入されます

19.1.1.2 XSLT 2.0

このセクション:

- エンジン適合性
- 下位互換性
- 名前空間
- スキーマ認識
- 実装固有の振る舞い

適合性

MapForce のXSLT 2.0 エンジンは、World Wide Web Consortium (ワールド・ワイド・ウェブ・コンソーシアム) (W3C) の2007 年 1月 23日版の XSLT 2.0 勧告 および2010 年 12月 14日版の XPath 2.0 勧告に準拠します。

下位互換性

XSLT 2.0 エンジンは下位互換性を有します。XSLT 2.0 エンジンの下位互換性が有効になるのは、XSLT 1.0 スタイルシートを処理するためにXSLT 2.0 エンジン が使用される際です。XSLT 1.0 エンジンと下位互換性を持つXSLT 2.0 エンジンにより作成される出力に違いがあるかもしれないことに注意してください。

名前空間

XSLT 2.0 スタイルシートは、XSLT 2.0. プレフィックス内で使用することのできる型コンストラクタおよび関数を使用するため、以下の名前空間を宣言する必要があります。下のリストは通常使用されるリストです。希望する場合は、代替プレフィックスを使用することもできます。

名前空間	プレフィックス	名前空間 URI
XML スキーマ型	`xs:`	`http://www.w3.org/2001/XMLSchema`
XPath 2.0 関数	`fn:`	`http://www.w3.org/2005/xpath-functions`

通常これらの名前空間は、以下のリストで表示されるように`xsl:スタイルシート`または、`xsl:transform` 要素で宣言されます:

```
<xsl:スタイルシート 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:スタイルシート>
```

次の点に注意してください:

- XSLT 2.0 エンジンは、(上のテーブルでリストされている)XPath 2.0 およびXQuery 1.0 関数 名前空間 を**デフォルトの関数名前空間**として使用します。XPath 2.0 およびXSLT 2.0 関数をプレフィックス無しでスタイルシート内で使用することができます。XPath 2.0 関数 名前空間 をスタイルシート内でプレフィックスと共に宣言すると、割り当てられた宣言内でプレフィックスを追加して使用することができます。t
- XML スキーマ名前空間から型コンストラクタ型を使用する場合、名前空間 宣言内で使用された、プレフィックスを使用して型コンストラクタを呼び出さなければなりません (例えば、`xs:date`)。
- XPath 2.0 関数の一部は、XML スキーマデータ型と同じ名前を保有します。例えば、XPath 関数 `fn:string` および `fn:boolean` のためには、同じローカル名 :`xs:string` および`xs:boolean` を持つ XML スキーマデータ型 が存在します。ですから、XPath 式 `string('Hello')`を使用する場合、式は、`xs:string('Hello')`ではなく`fn:string('Hello')` として検証します。

スキーマ認識
XSLT 2.0 エンジンは、スキーマを認識します。ですから、ユーザー定義 スキーマ型 および`xsl:validate` 命令を使用することができます。

実装固有の振る舞い
以下は、XSLT 2.0 エンジンが、特定のXSLT 2.0 関数の振る舞いの実装 特定のアスペクトをどのように扱うかの説明です。

`xsl:result-document`
追加してサポートされるエンコードは以下の通りです (Altova-固有):`x-base16tobinary` および`x-base64tobinary`.

`function-available`
インスコープ関数の使用をテストする関数 (XSLT、XPath、および拡張関数).

`unparsed-text`
`href` 属性は、以下を受け入れます (i) ベースuri フォルダー内のファイルの相対パス、および (ii) 相対パスを持つまたは持たない`file://` プロトコール。追加してサポートされるエンコードは以下の通りです (Altova-固有):`x-binarytobase16` および`x-binarytobase64`.

`unparsed-text-available`
`href` 属性は、以下を受け入れます (i) ベースuri フォルダー内のファイルの絶対パス、および (ii) 絶対パスを持つまたは持たない`file://` プロトコール。追加してサポートされるエンコードは以下の通りです (Altova-固有):`x-binarytobase16` および`x-binarytobase64`.

メモ: RaptorXML の先行製品であるAltovaXML で実装されていた以下のエンコード値は使用しないでください: `base16tobinary`, `base64tobinary`, `binarytobase16` and `binarytobase64`.

19.1.1.3 XQuery 1.0

このセクション:

- エンジン適合性
- スキーマ認識
- エンコード
- 名前空間
- XML ソースと検証
- 静的および動的な型のチェック
- ライブラリモジュール
- 外部モジュール
- 照合順序
- 数値データの精度
- XQuery 命令サポート

適合性

MapForce のXQuery 1.0 エンジンは、World Wide Web Consortium (ワールド・ワイド・ウェブ・コンソーシアム) (W3C) の2010 年 12月 14日版の XQuery 1.0 勧告に準拠します。XQuery 標準は、多数の機能の実装についての裁量を提供します。下には、XQuery 1.0 エンジンがどのようにこれらの機能を実装するかについて説明するリストが下に挙げられています。

スキーマ認識

XQuery 1.0 エンジンはスキーマを認識します。

エンコード

UTF-8 およびUTF-16 文字のエンコードは、サポートされています。

名前空間

以下の名前空間 URI と関連するバインドはは定義済みです。

名前空間	プレフィックス	名前空間 URI
XML スキーマ型	`xs:`	`http://www.w3.org/2001/XMLSchema`
スキーマインスタンス	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
内蔵の 関数	`fn:`	`http://www.w3.org/2005/xpath-functions`
Local 関数	`local:`	`http://www.w3.org/2005/xquery-local-functions`

次の点に注意してください:

- XQuery 1.0 エンジンは、上にリストされたプレフィックスを名前空間に対応するバウンドとして認識します。
- Since the 上にリストされた内蔵の関数 名前空間は、XQuery内のデフォルトの関数です。内蔵の関数 が呼び出される際、名前空間、fn:プレフィックス を使用する必要はありません。 (例えば、`string("Hello")` が`fn:string` 関数を呼び出す場合。) しかし、プレフィックス`fn:` は、クエリプロローグ内で名前空間を宣言することなく内蔵の関数を呼び出す時に使用することができます。 (サンプル:`fn:string("Hello")`).
- クエリプロローグ内で`default function` 名前空間 式を宣言することにより、デフォルトの関数 名前空間をすることにより変更することができます。
- XML スキーマ名前空間空の型を使用する場合、プレフィックス`xs:` は、名前空間を明確に宣言することなく、また、これらのプレフィックスをクエリプロローグ内でバインドすることなく使用することができます。 (サンプル:`xs:date` および`xs:yearMonthDuration`。) XML スキーマ名前空間のために、他のプレフィックスを使用する場合は、クエリプロローグ内で明確に宣言されている必要があります。 (サンプル:`declare 名前空間 alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- `untypedAtomic`, `dayTimeDuration`, および`yearMonthDuration` データ型 が 23January 2007 のCR 共に、XPath データ型 名前空間からXML スキーマ名前空間へ移動されていることに注意してください。ですから以下となります :`xs:yearMonthDuration`。

関数のための名前空間、型コンストラクタ ノードテスト、が間違って割り当てられている場合、エラーが発生します。しかし、一部の関数はスキーマデータ型と同じ名前を持つことに注意してください。 例 :`fn:string` および `fn:boolean`。(`xs:string` および`xs:boolean` は宣言されています)名前空間 プレフィックス 関数または 型コンストラクタが使用されるか決定します。

XML ソースドキュメントと検証

XML ドキュメントused in executing an XQuery ドキュメントwith XQuery 1.0 エンジンmust be 整形式 . しかし、they do not need to be valid according to an XML スキーマ.If the ファイルis not valid, the invalid ファイルis loaded without schema information. XML ファイルが外部スキーマと関連付けられ、また有効な場合、then post-schema 検証情報 is generated for XML データand will be used for クエリ検証 .

静的および動的な型のチェック

静的分析フェーズchecks aspects of クエリsuch as syntax, whether 外部レファレンス (例 for モジュール) exist, whether invoked 関数と変数 are defined, and so on. If an error is detected in 静的分析フェーズ, it is reported and the execution is stopped.

動的な型チェックis carried out ランタイム中 ,when クエリis actually executed. If a type is incompatible with the requirement of an operation, an error is reported. 例えば、式 `xs:string("1") + 1` は、エラーを返します。because the addition operation cannot be carried out on an operand of type `xs:string`.

ライブラリモジュール

ライブラリモジュールstore 関数と変数 so they can be reused. XQuery 1.0 エンジンsupports モジュールthat are stored in **a single external XQuery ファイル**. Such モジュール ファイルmust contain モジュール 宣言 in its prolog, which associates a target 名前空間 .以下はモジュールサンプルです :

```
module 名前空間 libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

すべての関数および変数は、モジュールに関連した名前空間に属するモジュール内で宣言されています。.モジュールis used by importing it into an XQuery ファイルwith the `import module` statement クエリプロローグ内の . `import module` ステートメントは、ライブラリモジュールファイル内で直接宣言された、関数と変数のみをインポートします。例 :

```
import module namespace modlib = "urn:module-library" at
   "modulefilename.xq";
if  ($modlib:company = "Altova")
then        modlib:webaddress()
else        error("No match found.")
```

外部関数

外部関数 は、サポートされていません。i.e. in those 式 using the `external` keyword, as in:

```
declare function hoo($param as xs:integer) as xs:string external;
```

照合順序
デフォルトの照合順序 is the Unicode-コードポイント照合順序 ,which compares strings on the basis of their Unicode コードポイント .その他にサポートされる照合順序 are the ICU 照合順序 listed here. To use a specific 照合順序 ,supply its URI as given in the list of supported 照合順序. Any string comparisons, including for the `fn:max` and `fn:min` 関数 ,will be made according to the specified 照合順序 .照合順序オプションが指定されていない場合、デフォルトのUnicode-コードポイント照合順序 が使用されます。

数値データの精度

- The `xs:integer` データ型 is arbitrary-精度 ,i.e. it can represent any number of 桁 .
- The `xs:decimal` データ型 has a limit of 20 桁 after the decimal point.
- The `xs:float` and `xs:double` データ型 have limited-精度 of 15 桁 .

XQuery 命令サポート
`Pragma` 命令 は、サポートされていません。発生した場合、無視されフォールバックの式が検証されます。

19.1.2 XSLT と XPath/ XQuery 関数

このセクションでは、XPath および/または XQuery 式で使用することのできる、Altova 拡張関数と他の拡張関数をリストします。Altova 拡張関数は Altova の XSLT および XQuery エンジンで使用することができ、W3C 標準で定義された関数ライブラリで使用することのできる機能に追加して機能を提供します。

一般的な情報
以下の一般的な情報に注意してください:

- W3C 仕様により定義されているコア関数ライブラリの関数は、関数の呼び出しにプレフィックスは必要ありません。これは、XSLT および XQuery エンジンが、 XPath/XQuery 関数仕様で指定されている `http://www.w3.org/2005/xpath-functions` プレフィックス無しの関数をデフォルト関数の名前空間に属するものとして読み込むためです。
- 関数において、各アイテムが引数となるようなシーケンスが期待されており、2つ以上のアイテムがシーケンスにより呼び出された場合、エラーが返されます。
- 全ての比較は Unicode コードポイントコレクションを使用することで行われます。
- QName の結果は `[prefix:]localname` という形式でシリアル化されます。

xs:decimal の精度

精度とは、数値内にある桁数のことで、仕様では少なくとも18桁が求められます。`xs:decimal` 型に結果が収められる除算の場合、端数処理を行うことなく精度は小数点以下の19桁になります。

黙示的なタイムゾーン
2つの`date`、`time`、または`dateTime` 値を比較する場合、比較する値のタイムゾーンを明らかにする必要があります。値の中にタイムゾーンが明示的に与えられていない場合、黙示的なタイムゾーンが使用されます。黙示的なタイムゾーンはシステムクロックから取得され`implicit-timezone()` 関数によりその値をチェックすることができます。

照合順序
デフォルトの照合順序は、Unicode コードポイントをベースに文字列を比較するUnicode コードポイント照合順序です。エンジンはUnicode 照合アルゴリズムを使用しています。他のサポートされる照合順序は下にリストされるICU 照合順序 です。使用するには、サポートされる照合順序のリストのURI を提供してください (*下のテーブル*)。`max` と`min` 関数を含む、文字列の比較は、指定された照合順序に沿って行われます。照合順序オプションが指定されていない場合、デフォルトのUnicode コードポイント照合順序が使用されます。

言語	URI
`da`: デンマーク語	`da_DK`
`de`: ドイツ語	`de_AT, de_BE, de_CH, de_DE, de_LI, de_LU`
`en`: 英語	`en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW`
`es`: スペイン語	`es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE`
`fr`: フランス語	`fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG`
`it`: イタリア語	`it_CH, it_IT`
`ja`: 日本語	`ja_JP`
`nb`: ノルウェー語 (ブークモール)	`nb_NO`
`nl`: オランダ語	`nl_AW, nl_BE, nl_NL`
`nn`: ノルウェー語 (ニーノシュク)	`nn_NO`
`pt`: ポルトガル語	`pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST`
`ru`: ロシア語	`ru_MD, ru_RU, ru_UA`
`sv`: スウェーデン語	`sv_FI, sv_SE`

名前空間軸
名前空間軸はXPath 2.0 にて廃止されましたが、名前空間軸の使用はサポートされています。XPath 2.0 *メカニズム*により名前空間情報へアクセスするには、`in-scope-prefixes()`、`namespace-uri()`、`namespace-uri-`

for-prefix(関数を使用してください。

19.1.2.1 Altova 拡張関数

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、** `http://www.altova.com/xslt-extensions` に収められており、`altova:` プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることに注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

W3C のXPath/XQuery 関数仕様で定義された関数は、以下で使用することができます：(i) XSLT 子アンテキスト内のXPath 式と (ii) XQuery 文書内のXQuery 式。このドキュメントでは、前者 (XSLT 内のXPath) のコンテキストで使用することのできる関数を、`XP` シンボルと共に表示し、と称します。後者 (XQuery) で使用することのできる関数はの`XQ` シンボルと共に表示され、XQuery 関数と共に作業することができます。W3C のXSLT 仕様は、XPath/XQuery 関数の仕様ではなく、XSLT 文書内のXPath 式でも使用することのできる関数を定義します。これらの関数は、`XSLT` シンボルと共に表示され、XSLT 関数と称されます。関数を使用することのできるXPath/XQuery およびXSLT のバージョンは、関数の詳細に記載されています (*下のシンボルを参照してください*)。XPath/XQuery およびXSLT 関数ライブラリからの関数は、*プレフィックス*無しでリストされています。Altova 拡張関数などの、他のライブラリからの関数は*プレフィックス*と共にリストされています。

XPath 関数 (XSLT 内のXPath 式で使用):	`XP1` `XP2` `XP3`
XSLT 関数 (XSLT 内のXPath 式で使用):	`XSLT1` `XSLT2` `XSLT3`
XQuery 関数 (XQuery 内のXQuery 式で使用):	`XQ1` `XQ3`

XSLT 関数
XSLT 関数はXSLT 2.0 の`current-group()` や`key()` 関数と同様に、XSLT コンテキストにて使用することができます。 (例えば、XQuery コンテキストなどの)非 XSLT コンテキストでは使用することができません。XBRL に対するXSLT 関数は、XBRL をサポートするエディションのAltova 製品でのみ使用することができます。

XPath/XQuery 関数
XPath/XQuery 関数は、 XSLT コンテキスト、 XQuery 関数のXPath 式で使用することができます：

- 日付 時刻
- 位置情報
- イメージに関連した
- 数値
- シーケンス
- 文字列
- その他

チャート関数 (Enterprise および Server Editions のみ)
チャート関数のためのAltova 拡張子は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていません。XML データからチャートを生成することができます。

XSLT 関数

XSLT 拡張関数 はXSLT コンテキスト内のXPath 式にて使用することができます。 (例えば、XQuery コンテキストなどの)非 XSLT コンテキストでは使用することができません。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、** `http://www.altova.com/xslt-extensions` に収められており、`altova:` プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	XP1 XP2 XP3
XSLT 関数 (XSLT 内のXPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内のXQuery 式で使用):	XQ1 XQ3

標準関数

distinct-nodes [altova:]

altova:distinct-nodes(*node()**)** を**node()*** とする XSLT1 XSLT2 XSLT3
入力として1つ以上のノードを必要とし、同じセットから重複した値を持つノードを除いたノードを返します。XPath/XQuery 関数 `fn:deep-equal` を使用して比較を行うことができます。

サンプル

- **altova:altova:distinct-nodes**(`country`)は重複した値を持つものを除く全ての子 `country` ノード返します。

evaluate [altova:]

altova:evaluate(XPathExpression *as xs:string***[, ValueOf$p1, ... ValueOf$pN])**
XSLT1 XSLT2 XSLT3
XPath 式を必要とし、必須引数として文字列をパスします。評価された式の出力を返します。例えば:
altova:evaluate('//Name[1]') は、ドキュメント内の最初の `Name` 要素のコンテンツを返します。式 `//Name[1]` は、一重引用符を使用することにより、文字列としてパスされます。

`altova:evaluate` 関数は、オプションとして追加の引数を持つことができます。これらの引数は、`p1`, `p2`, `p3`... `pN` の名前を持つスコープ内の変数の値です。使用に関して以下の点に注意してください: (i) 変数は、`X` が整数である箇所のフォーム `pX` の名前と共に定義される必要があります。(ii) `altova:evaluate` 関数の引数は、(*上の署名参照*) 2 番目の引数からは、数値順の変数のシーケンスに対応した引数のシーケンス変数の値を与えます: `p1` to `pN`: 第 2引数は変数 `p1` の値で、第 3引数は、変数 `p2` の値です。 (ii) 変数の値は型 `item*` である必要があります。

サンプル

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

outputs `"hi 20 10"`

上のリストに関して、以下の点に注意してください：

- altova:evaluate 式の第 2引数は、変数 $p1 に割り当てられた値で、第三の引数は変数 $p2 に割り当てられた値です。
- 関数の第 4番目の引数は、引用符による囲いで表示された文字列の値です。
- xs:variable 要素の select 属性は、XPath 式を提供します。この式は xs:string の型である必要があり、一重引用符で囲まれています。

変数の使用方法を更に説明するサンプル

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  *最初の*Name *要素の出力値*

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```
  Outputs "//Name[1]*"*

altova:evaluate() 拡張関数は、XSLT スタイルシート内のXPath 式が動的に評価される必要のある1つ以上の部分を持つシチュエーションで役に立ちます。例えば、ユーザーが並べ替えの必要条件をリクエストする場合、このシチュエーションは属性 UserReq/@sortkey に保管されます。スタイルシートでは、以下の式が使用できます：<xsl:sort select="**altova:evaluate(../UserReq/@sortkey)**" order="ascending"/>。altova:evaluate() 関数は、コンテキストノードの親の UserReq 子要素の sortkey 属性を読み込みます。sortkey 属性の値が Price の場合、Price は、altova:evaluate() 関数により返され、select 属性 <xsl:sort select="**Price**" order="ascending"/> の値になります。この sort 命令が、Order という要素のコンテキスト内で発生する場合、Order 要素は Price の子の値に従い並べ替えられます。また、@sortkey の値が、Date の場合、Order 要素は、Date の子の値に従い並べ替えられます。ですから、Order の並べ替えの条件は、ランタイムでの sortkey 属性から選択されます。これは、以下の式などでは達成することはできません：<xsl:sort select="**../UserReq/@sortkey**" order="ascending"/>。上の場合、並べ替え条件は sortkey 属性自身であり、Price または Date（または、現在の sortkey のコンテンツ）ではありません。

メモ： 静的なコンテキストは、呼び出し環境の名前空間、型、機能、しかし変数を以外を含みます。ベースURI とデフォルトの名前空間は継承されます。

追加サンプル

- 静的な変数：<xsl:value-of select="$i3, $i2, $i1" />
  3 つの変数の値を出力します。

- 動的な変数を持つ動的 XPath 式：
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />
  ```
  "30 20 10" *を出力します。*

- 動的な変数を持たないXPath 式：
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath)" />
  ```
  出力エラー：$p3 に対して定義されている変数はありません。

encode-for-rtf [altova:]

```
altova:encode-for-rtf(input as xs:string, preserveallwhitespace as
```

xs:boolean, **preservenewlines** *as xs:boolean*) を**xs:string** とする XSLT2 XSLT3
RTF のためのコードに入力文字列を変換します。空白と新しい行は、それぞれの引数により指定されるboolean の値に基づき保管されます。

[トップ]

XBRL 関数

Altova XBRL 関数はXBRL をサポートするAltova 製品のエディションのみで使用することができます。

xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** を**node()*** とする XSLT2 XSLT3
ノードを入力引数として必要とし、入力ノードに参照されるXBRL フットノート ノードを返します。

xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName*, *xs:string***)** を**node()*** とする XSLT2 XSLT3
以下の 2 つの入力引数を必要とします : ノード名とノードを含むタクソノミファイルロケーション。関数は、入力ノードと関連したXBRL ラベルノードを返します。

[トップ]

XPath/ XQuery 関数 :日付と時刻

Altova の日付 時刻拡張関数はXPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	XP1 XP2 XP3
XSLT 関数 (XSLT 内のXPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内のXQuery 式で使用):	XQ1 XQ3

機能によりグループ化

- xs:dateTime に期間を追加して、xs:dateTimeを返す
- xs:date に期間を追加して、xs:date を返す
- xs:time に期間を追加して、return xs:time を返す
- フォーマットと期間の取得

- 現在の日付 時刻を生成する関数からタイムゾーンを削除する
- 日付から整数を週の曜日として返す
- 日付から週数を整数として返す
- 各型の構文コンポーネントから日付、時刻、期間 の型を構築する
- 文字列入力から日付、日付時刻 または時刻 を構築する
- 年齢に関連した関数

▼ アルファベット順にグループ化

altova:add-days-to-date
altova:add-days-to-dateTime
altova:add-hours-to-dateTime
altova:add-hours-to-time
altova:add-minutes-to-dateTime
altova:add-minutes-to-time
altova:add-months-to-date
altova:add-months-to-dateTime
altova:add-seconds-to-dateTime
altova:add-seconds-to-time
altova:add-years-to-date
altova:add-years-to-dateTime
altova:age
altova:age-details
altova:build-date
altova:build-duration
altova:build-time
altova:current-dateTime-no-TZ
altova:current-date-no-TZ
altova:current-time-no-TZ
altova:format-duration
altova:parse-date
altova:parse-dateTime
altova:parse-duration
altova:parse-time
altova:weekday-from-date
altova:weekday-from-dateTime
altova:weeknumber-from-date
altova:weeknumber-from-dateTime

[トップ]

xs:dateTime に期間を追加する XP3 XQ3

これらの関数は **xs:dateTime** に期間を追加し、**xs:dateTime** を返します。xs:dateTime 型はCCYY-MM-DDThh:mm:ss.sss のフォーマットです。これはxs:date とxs:time フォーマットの連結で、T により区切られています。タイムゾーンサフィックス+01:00 (for example)は任意です。

▼ add-years-to-dateTime [altova:]

altova:add-years-to-dateTime(DateTime *as xs:dateTime*, **Years** を *xs:integer*) as **xs:dateTime** とする XP3 XQ3

日付までの期間を年数で表示します。第 2の引数は第 1の引数として与えられたxs:date に追加される年数です。結果はxs:date 型です。

▫ *サンプル*

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) returns 2024-01-15T14:00:00
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4)

returns `2010-01-15T14:00:00`

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(DateTime *as xs:dateTime*, **Months** を *xs:integer*) as **xs:dateTime** とする **XP3 XQ3**

xs:dateTime に月数での期間を追加します (*下のサンプル参照*)。第 2の引数は、第 1の引数として与えられたxs:dateTime に追加される月数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) 2014-11-15T14:00:00 を返します。
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) 2013-11-15T14:00:00 を返します。

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(DateTime *as xs:dateTime*, **Days** *as xs:integer*) を **xs:dateTime** とする **XP3 XQ3**

xs:dateTime に日数での期間を追加します (*下のサンプル参照*)。第 2の引数は、第 1の引数として与えられたxs:dateTime に追加される日数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10)は 2014-01-25T14:00:00 を返します。
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8)は 2014-01-25T14:00:00 を返します。

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(DateTime *as xs:dateTime*, **Hours** *as xs:integer*) を**xs:dateTime** とする **XP3 XQ3**

xs:dateTime に時間数での期間を追加します (*下のサンプル参照*)。第 2の引数は、第 1の引数として与えられたxs:dateTime に追加される時間数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) は 2014-01-15T23:00:00 を返します。
- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) は 2014-01-15T05:00:00 を返します。

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(DateTime *as xs:dateTime*, **Minutes** *as xs:integer*) を**xs:dateTime** とする **XP3 XQ3**

xs:dateTime に分数での期間を追加します (*下のサンプル参照*)。第 2の引数は、第 1の引数として与えられたxs:dateTime に追加される分数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) 2014-01-15T14:55:00 を返します。
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5)

2014-01-15T14:05:00 を返します。

add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(DateTime *as xs:dateTime*, **Seconds** を *xs:integer*) as **xs:dateTime** とする XP3 XQ3

xs:dateTime に秒数での期間を追加します (*下のサンプル参照*)。第 2の引数は、第 1の引数として与えられたxs:dateTime に追加される秒数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20)
  2014-01-15T14:00:30 を返します。
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5)
  2014-01-15T14:00:05 を返します。

[トップ]

---

xs:date に期間を追加する XP3 XQ3

これらの関数は**xs:date** に期間を追加し、**xs:date** を返します。xs:date 型はCCYY-MM-DD フォーマットです。

add-years-to-date [altova:]

altova:add-years-to-date(Date *as xs:date*, **Years** *as xs:integer*) を**xs:date** とする XP3 XQ3

日付までの期間を年数で表示します。第 2の引数は第 1の引数として与えられたxs:date に追加される年数です。結果はxs:date 型です。

サンプル

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) は2024-01-15 を返します。
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) は2010-01-15 を返します。

add-months-to-date [altova:]

altova:add-months-to-date(Date *as xs:date*, **Months** *as xs:integer*) を**xs:date** とする XP3 XQ3

日付までの期間を月数で表示します。第 2の引数は第 1の引数として与えられたxs:date に追加される月数です。結果はxs:date 型です。

サンプル

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) 2014-11-15 を返します。
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) 2013-11-15 を返します。

add-days-to-date [altova:]

altova:add-days-to-date(Date *as xs:date*, **Days** *as xs:integer*) を**xs:date** とする XP3 XQ3

日付までの期間を日数で表示します。第 2の引数は第 1の引数として与えられたxs:date に追加される日数です。結果はxs:date 型です。

サンプル

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10)は 2014-01-25 を返します。
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) は2014-01-07 を返します。

[トップ]

フォーマットと期間の取得 XP3 XQ3

これらの関数は**xs:date** に期間を追加し、**xs:date** を返します。xs:date 型はCCYY-MM-DD フォーマットです。

format-duration [altova:]

altova:format-duration(Duration *as xs:duration*, **Picture** *as xs:string*) as **xs:string** とする XP3 XQ3

第 1の引数として提出された期間を、第 2の引数として提出された文字列によりをフォーマットします。出力は、文字列によりフォーマットされたテキスト文字列です。

サンプル

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") は"Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7" を返します。
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") は"Months:03 Days:02 Hours:02 Minutes:53" を返します。

parse-duration [altova:]

altova:parse-duration(InputString *as xs:string*, **Picture** *as xs:string*) を **xs:duration** とする XP3 XQ3

パターン化された文字列を最初の引数として、文字を第 2の引数とします。入力文字列は文字をベースに解析され、xs:duration が返されます。

サンプル

- **altova:parse-duration**("Days:02 Hours:02 Minutes:53 Seconds:11

Fractions:7"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") は"P2DT2H53M11.7S" を返します。

- **altova:parse-duration**("Months:03 Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7", "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") は"P3M2DT2H53M" を返します。

[Top]

xs:time に期間を追加する XP3 XQ3

これらの関数はxs:time に期間を追加し、 xs:time を返します。 xs:time 型はhh:mm:ss.sss 構文フォームです。

文字 z は協定世界時 (UTC) を表します。他のタイムゾーンは、UTC との差異を+hh:mm または -hh:mm のフォーマットで表示しています。タイムゾーンの値が無い場合は、UTC ではない未知のタイムゾーンとして見なされます。

add-hours-to-time [altova:]

altova:add-hours-to-time(Time *as xs:time*, **Hours** *as xs:integer*) を**xs:time** とする XP3 XQ3

日付までの期間を時間数で表示します。第 2の引数は第 1の引数として与えられたxs:time に追加される時間数です。結果はxs:time 型です。

サンプル

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) は21:00:00 を返します。
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) は04:00:00 を返します。

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) を**xs:time** とする XP3 XQ3

日付までの期間を分数で表示します。第 2の引数は第 1の引数として与えられたxs:date に追加される分数です。結果はxs:date 型です。

サンプル

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) 14:55:00 を返します。
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) 14:05:00 を返します。

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) を**xs:time** とする XP3 XQ3

時間までの期間を秒数で表示します。第 2の引数は、第 1の引数として与えられたxs:time に追加される秒数です。結果はxs:time 型です。第 2のコンポーネントは0 から59.999 の範囲であることができます。

サンプル

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) は14:00:20 を返します。
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) は14:00:20.895 を返します。

[Top]

現在の日付 /時刻を生成する関数からタイムゾーンを削除する XP3 XQ3

これらの関数は、現在の**xs:dateTime**、**xs:date**、または**xs:time** 値からそれぞれタイムゾーンを削除します。xs:dateTime とxs:dateTimeStamp の差異は、後者のタイムゾーンが必要な場合です。前者の場合は任意です。) xs:dateTimeStamp 値のフォーマットはCCYY-MM-DDThh:mm:ss.sss±hh:mm またはCCYY-MM-DDThh:mm:ss.sssZ です。、日付と時刻がxs:dateTimeStamp としてシステムクロックから読み込まれる場合、current-dateTime-no-TZ() 関数がタイムゾーンを削除するために使用されます。

current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() を**xs:dateTime** とする XP3 XQ3

この関数は引数を必要としません。current-dateTime() (システムクロックによる現在の時刻)のタイムゾーンの部分を削除し、xs:dateTime の値を返します。

サンプル

現在のdateTime が**2014-01-15T14:00:00+01:00** の場合：

- **altova:current-dateTime-no-TZ**() は2014-01-15T14:00:00 を返します。

current-date-no-TZ [altova:]

altova:current-date-no-TZ() を**xs:date** とする XP3 XQ3

この関数は引数を必要としません。current-date() (システムクロックによる現在の時刻)のタイムゾーンの部分を削除し、xs:date の値を返します。

サンプル

現在のdate が**2014-01-15+01:00** の場合：

- **altova:current-date-no-TZ**() は2014-01-15 を返します。

current-time-no-TZ [altova:]

altova:current-time-no-TZ() を**xs:time** とする XP3 XQ3

この関数は引数を必要としません。current-time() (システムクロックによる現在の時刻)のタイムゾーンの部分を削除し、xs:time の値を返します。

サンプル

現在のtime が**14:00:00+01:00** の場合：

- **altova:current-time-no-TZ**() は14:00:00 を返します。

[Top]

---

xs:dateTime または xs:date として曜日を返す XP3 XQ3

これらの関数は、xs:dateTime またはxs:date から曜日を整数として返します。曜日は (米国式フォーマットを使用して)1 から7 と番号づけられています。 日曜=1 と番号付けられます。欧州のフォーマットでは月曜 (=1) として番号付けられます。日曜=1 である米国フォーマットは整数 0 がフォーマットを表示するために使用できる箇所で設定することができます。

weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(DateTime *as xs:dateTime*) を**xs:integer** とする XP3 XQ3

時刻付きの日付を単一の引数として、この日付の曜日を正数として返します。曜日は 日曜=1 から開始して番号を

付けます。(月曜=1 とする) ヨーロッパのフォーマットが必要な場合、この関数の他の署名を使用します (*下の次の署名を参照*)。

サンプル

- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"))` は月曜を表示する`2` を返します。

altova:weekday-from-dateTime(DateTime *as xs:dateTime*, **Format** *as xs:integer*) as **xs:integer** とする **XP3 XQ3**

時間月の日付を最初の引数として、この日付の曜日を正数として返します。曜日は月曜=1 から開始して番号を付けます。第 2の引数 (整数) が0 の場合、日曜=1 から開始して、曜日は1から7 と番号を付けます。第 2の引数が0 以外の整数の場合、月曜=1です。第 2の引数がない場合、関数はこの関数の他の署名を持つと読み込まれます (*前の次の署名を参照*)。

サンプル

- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 1)` は月曜を表示する`1` を返します。
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 4)` は月曜を表示する`1` を返します。
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 0)` は月曜を表示する`2` を返します。

weekday-from-date [altova:]

altova:weekday-from-date(Date *as xs:date*) を **xs:integer** とする **XP3 XQ3**

日付を単一の引数として、この日付の曜日を正数として返します。曜日は日曜=1 から開始して番号を付けます。(月曜=1 とする) ヨーロッパのフォーマットが必要な場合、この関数の他の署名を使用します (*下の次の署名を参照*)。

サンプル

- `altova:weekday-from-date(xs:date("2014-02-03+01:00"))` は月曜を表示する`2` を返します。

altova:weekday-from-date(Date *as xs:date*, **Format** *as xs:integer*) を **xs:integer** とする **XP3 XQ3**

日付を最初の引数とし、この日付の曜日を正数として返します。曜日は月曜=1 から開始して番号を付けます。第 2 (フォーマット)引数が0,の場合、日曜=1 から開始し、曜日は1 から7で番号付けられます。第 2 引数が整数で 0 以外の場合、月曜=1です。第 2 の引数がない場合、関数はこの関数の他の署名を持つと読み込まれます (*前の署名を参照*)。

サンプル

- `altova:weekday-from-date(xs:date("2014-02-03"), 1)` は月曜を示す`1` を返します。
- `altova:weekday-from-date(xs:date("2014-02-03"), 4)` は月曜を示す`1` を返します。
- `altova:weekday-from-date(xs:date("2014-02-03"), 0)` は月曜を示す`2` を返します。

[Top]

xs:dateTime または xs:date から週数を返す XP2 XQ1 XP3 XQ3

これらの関数は週数 (整数として) を xs:dateTime から xs:date. から返します。週の番号付けは、米国、欧州、イスラムのカレンダーフォーマットで使用することができます。週の始まりが異なるため、週数の番号付けは、カレンダーのフォーマットにより異なります。(米国フォーマットでは、日曜、欧州フォーマットでは月曜、イスラムフォーマットでは土曜が週の開始日です。)

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Date *as xs:date*, **Calendar** *as xs:integer*) を **xs:integer** とする XP2 XQ1 XP3 XQ3

正数として提出された **Date** 引数の週数を返します。第 2 の引数 (**カレンダー**) は続くカレンダーシステムを指定します。

サポートされる **カレンダー** の値は次のとおりです :

- **0 = US 米国のカレンダー** (週の始まりは日曜日)
- **1 = ISO 標準、 欧州のカレンダー** (週の始まりは月曜日)
- **2 = イスラムのカレンダー** (週の始まりは土曜日)

デフォルトは **0** です。

サンプル

- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 0) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 1) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 2) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23")) は 13 を返します。

上のサンプル (2014-03-23) の date の曜日は日曜日です。米国およびイスラムのカレンダーは欧州カレンダーのこの日付よりも一週間先です。

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(DateTime *as xs:dateTime*, **Calendar** *as xs:integer*) を **xs:integer** とする XP2 XQ1 XP3 XQ3

正数として提出された **DateTime** 引数の週数を返します。第 2 の引数 (**カレンダー**) は続くカレンダーシステムを指定します。

サポートされる **カレンダー** の値は次のとおりです :

- **0 = US 米国のカレンダー** (週の始まりは日曜日)
- **1 = ISO 標準、 欧州のカレンダー** (週の始まりは月曜日)
- **2 = イスラムのカレンダー** (週の始まりは土曜日)

Default is **0**.

サンプル

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 0) は 13 を返します。
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 1) は 13 を返します。
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 2) は 13 を返します。
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00")) は 13 を返します。

上のサンプル (2014-03-23T00:00:00) の dateTime の曜日は日曜日です。米国およびイスラムのカレンダーは欧州カレンダーのこの日付よりも一週間先です。

[トップ]

各型の構文コンポーネントから日付、時刻、期間の型を構築する XP3 XQ3

関数は、xs:date, xs:time または xs:duration の構文コンポーネントを入力引数とし、引数を結合させて対応するデータ型を構築します。

build-date [altova:]

altova:build-date(Year *as xs:integer*, **Month** *as xs:integer*, **Date** *as xs:integer*) を **xs:date** とする XP3 XQ3

第 1、第 2、第3の引数は、それぞれ、年、月、日を表します。xs:date 型の値を構築するため結合されます。整数の値は、特定の日付の一部の適正な範囲内である必要があります。例えば第 2の引数 (月の部分)は12 以上であってはなりません。

サンプル

- **altova:build-date**(2014, 2, 03) は2014-02-03 を返します。

build-time [altova:]

altova:build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*) を **xs:time** とする XP3 XQ3

第 1、第 2、第3引数はそれぞれ時間数 (0 から23)、分数 (0 から59)、および秒数 (0 から59) の値です。これらの値は、xs:time 型の値を作成するために結合されます。整数の値は、それぞれの部分の正しい範囲内である必要があります。例えば、秒 (分)引数は、59 以上であってはなりません。値にタイムゾーンを追加するには、この関数の他の署名を使用してください (*次の署名を参照*)。

サンプル

- **altova:build-time**(23, 4, 57) は23:04:57 を返します。

altova:build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*, **TimeZone** *as xs:string*) を **xs:time** とする XP3 XQ3

第 1、第 2、第 3引数はそれぞれ時間数 (0 から23)、分数 (0 から59)、および秒数 (0 から59) の値です。第四の引数は、値の一部としてタイムゾーンを与えます。 4つの引数が結合され、xs:time 型の値を構築します。例えば、第 2の引数 (分数) は59 以上であってはなりません。

サンプル

- **altova:build-time**(23, 4, 57, '+1') は23:04:57+01:00 を返します。

build-duration [altova:]

altova:build-duration(Years *as xs:integer*, **Months** *as xs:integer*) を **xs:yearMonthDuration** とする XP3 XQ3

xs:yearMonthDuration 型の値を構築するためには 2つの引数が必要です。最初の引数は期間の年数 値の部分を与え、第 2の引数は、月数 値の部分を与えます。第 2の引数 (月数)が同じまたはより大きくなると、整数は12 により分割されます。 商は、年数 の部分を表す 最初の引数に加算され、除算の残りは の部分は月数 の部分を表すために使用されます。期間のxs:dayTimeDuration 型を作成するには、次の署名を参照してください。

サンプル

- **altova:build-duration**(2, 10) はP2Y10M を返します。
- **altova:build-duration**(14, 27) はP16Y3M を返します。
- **altova:build-duration**(2, 24) はP4Y を返します。

altova:build-duration(Days *as xs:integer*, **Hours** *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*) を**xs:dayTimeDuration** とする XP3 XQ3
xs:dayTimeDuration 型の値を構築するためには 4 つの引数が必要です。最初の引数は期間の日数 値の部分で、第 2、第 3、第 4引数は、それぞれ期間の時間数、分数、秒数 値のです。これらの 3 つの時間 引数は、次の高い単位の値に加算され、結果は期間の全体の値を計算するために使用されます。例えば、 72秒は、1M+12S (1 分 と12秒)に変換され、この値が期間全体の値を計算するために使用されます。
xs:yearMonthDuration 型の期間を構築するためには、次の署名を参照してください。

サンプル

- **altova:build-duration**(2, 10, 3, 56) はP2DT10H3M56S を返します。
- **altova:build-duration**(1, 0, 100, 0) はP1DT1H40M を返します。
- **altova:build-duration**(1, 0, 0, 3600) はP1DT1H を返します。

[トップ]

文字列入力から日付、日付時刻 または 時刻 を構築する XP2 XQ1 XP3 XQ3

関数は、文字列を引数として、xs:date、xs:dateTime または xs:time データ型を構築します。文字列は、データ型のコンポーネントを提出されたパターン引数をベースにして分析されます。

parse-date [altova:]

altova:parse-date(Date *as xs:string*, **DatePattern** *as xs:string*) を**xs:date** とする XP2 XQ1 XP3 XQ3
Date 入力文字列を**xs:date** の値として返します。第二の引数である**DatePattern** は、入力文字列のパターン (コンポーネントのシーケンス)を指定します。**DatePattern** は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネント セパレータと共に説明されています。下のサンプルを参照してください。

D 日付
M 月
Y 年

DatePattern のパターンは、**Date** のパターンに一致する必要があります。出力は**xs:date** 型であるため、出力は常に **YYYY-MM-DD** 構文フォーマットになります。

サンプル

- **altova:parse-date**(xs:string("06-03-2014"), "[D]-[M]-[Y]") は2014-03-06 を返します。
- **altova:parse-date**(xs:string("06-03-2014"), "[M]-[D]-[Y]") は2014-03-06 を返します。
- **altova:parse-date**("06/03/2014", "[M]/[D]/[Y]") は2014-03-06 を返します。
- **altova:parse-date**("06 03 2014", "[M] [D] [Y]") は2014-03-06 を返します。
- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") は2014-03-06 を返します。

parse-dateTime [altova:]

altova:parse-dateTime(DateTime *as xs:string*, **DateTimePattern** *as xs:string*) を**xs:dateTime** とする XP2 XQ1 XP3 XQ3
DateTime 入力文字列を**xs:dateTime** の値として返します。第二の引数である**DateTimePattern** は、入力文字列のパターン (コンポーネントのシーケンス)を指定します。**DateTimePattern** は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネント セパレータと共に説明されています。下のサンプルを参照してください。

D	日
M	月
Y	年
H	時間
m	分
s	秒

DateTimePattern のパターンは、**DateTime** のパターンに一致する必要があります。出力は**xs:dateTime** 型であるため、出力は常に**YYYY-MM-DDTHH:mm:ss** 構文フォーマットになります。

サンプル

- **altova:parse-dateTime**(xs:string("06-03-2014 13:56:24"), "[D]-[M]-[Y] [H]:[m]:[s]") は2014-03-06T13:56:24 を返します。
- **altova:parse-dateTime**("time=13:56:24; date=06-03-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") は2014-03-06T13:56:24 を返します。

parse-time [altova:]

altova:parse-time(Time *as xs:string*, **TimePattern** *as xs:string***)** を**xs:time** とする **XP2 XQ1 XP3 XQ3**

Time 入力文字列を**xs:time** の値として返します。第二の引数である**TimePattern** は、入力文字列のパターン (コンポーネントのシーケンス) を指定します。**TimePattern** は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネントセパレータと共に説明されています。下のサンプルを参照してください。

H	時間
m	分
s	秒

TimePattern のパターンは、**Time** のパターンに一致する必要があります。出力は**xs:time** 型であるため、出力は常に**YYYY-HH:mm:ss** 構文フォーマットになります。

サンプル

- **altova:parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") は13:56:24 を返します。
- **altova:parse-time**("13-56-24", "[H]-[m]") は13:56:00 を返します。
- **altova:parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") は13:56:24 を返します。
- **altova:parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") は13:56:24 を返します。

[トップ]

年齢に関連した関数 XP3 XQ3

関数は計算された年齢 (i) 入力引数の日付と現在の日付の期間 (i) 2 つの入力引数の日付の期間 を返します。**altova:age** 関数は、年齢を年数で返します、**altova:age-details** は年齢を年数、月数、日数からなる3つの整数のシーケンスで返します。

age [altova:]

altova:age(StartDate *as xs:date*) を**xs:integer** とする XP3 XQ3
引数として提出された開始日からシステムクロックから取得された現在の日付 までの日数を数えて、あるオブジェクトの年齢の年数である正数を返します。入力引数が 1 年よりも大きい場合、または一年の場合、返される値は負の数です。

サンプル

If the current date is **2014-01-15**:

- **altova:age**(xs:date("2013-01-15")) は1 を返します。
- **altova:age**(xs:date("2013-01-16")) は0 を返します。
- **altova:age**(xs:date("2015-01-15")) は-1 を返します。
- **altova:age**(xs:date("2015-01-14")) は0 を返します。

altova:age(StartDate *as xs:date*, **EndDate** *as xs:date*) を**xs:integer** とする XP3 XQ3
最初の引数として提出された開始日から第 2 の引数の終了日 までの日数を数えて、あるオブジェクトの年齢の年数である正数を返します。最初の引数が 1 年または第 2 の引数よりも後の日付の場合、返される値は負の数です。

サンプル

If the current date is **2014-01-15**:

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) は10 を返します。
- **altova:age**(xs:date("2000-01-15"), current-date()) は、現在の日付が2014-01-15 の場合 14 を返します。
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) は-4 を返します。

age-details [altova:]

altova:age-details(InputDate *as xs:date*) を (**xs:integer**)* とする XP3 XQ3
引数とシステムクロックから取得された現在の日付として提出された日付の間の年数、月数、日数の 3 つの整数を返します。返されたyears+months+days の合計は、2 つの日付 (入力の日付と現在の日付) の時間差です。 入力の日付は現在の日付よりも早いまたは遅い値をもつことができますが、入力の日付が早いか遅いかは返される値で示されていません。戻される値は常に正の数です。

サンプル

現在の日付が**2014-01-15** の場合 :

- **altova:age-details**(xs:date("2014-01-16")) は(0 0 1)を返します。
- **altova:age-details**(xs:date("2014-01-14")) は(0 0 1) を返します。
- **altova:age-details**(xs:date("2013-01-16")) は(1 0 1) を返します。
- **altova:age-details**(current-date()) は(0 0 0)を返します。

altova:age-details(Date-1 *as xs:date*, **Date-2** *as xs:date*) を (**xs:integer**)* とする XP3 XQ3
二つの引数間の年数、月数、日数の 3 つの整数を返します。 返されたyears+months+days の合計は、2 つの入力の日付 の時間差です。最初の引数として提出される二つの日付はどちらが速くても、また遅くてもかまいません。返される値は入力の日付が現在の日付より早いかまた遅いかを示しません。戻される値は常に正の数です。

サンプル

- **altova:age-details**(xs:date("2014-01-16"), xs:date("2014-01-15")) は(0 0 1) を返します。
- **altova:age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) は(0 0 1)を返します。

[Top]

XPath/ XQuery 関数 :位置情報

以下の位置情報 XPath/XQuery 拡張関数は、MapForce の現在のバージョンによりサポートされています。また、次で使用することができます : (i) XSLT コンテキスト内の XPath 式、または (i) XQuery ドキュメント内の XQuery 式。

関数の名前指定と言語の適用性

Altova 拡張関数は XPath/XQuery 式で使用することができ、XPath、XQuery、および XSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は **Altova 拡張関数名前空間、`http://www.altova.com/xslt-extensions`** に収められており、**`altova:`** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意ください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内の XPath 式で使用):	**`XP1` `XP2` `XP3`**
XSLT 関数 (XSLT 内の XPath 式で使用):	**`XSLT1` `XSLT2` `XSLT3`**
XQuery 関数 (XQuery 内の XQuery 式で使用):	**`XQ1` `XQ3`**

`parse-geolocation [altova:]`

`altova:parse-geolocation(GeolocationInputString` *`as xs:string`***`)`** を **`xs:decimal+`** とする **`XP3` `XQ3`**

`GeolocationInputString` 引数を解析して、位置情報の緯度と経度 (この通りの順番) を 2 つの `xs:decimal` アイテムのシーケンスとして返します。位置情報入力文字列が提供されることのできるフォーマットは以下のリストの通りです。

メモ: `image-exif-data` 関数と Exif メタデータの `@Geolocation` 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **`altova:parse-geolocation`**`("33.33  -22.22")` は 2 つの `xs:decimals (33.33, 22.22)` のシーケンスを返します。
- **`altova:parse-geolocation`**`("48°51'29.6""N  24°17'40.2""")` は 2 つの `xs:decimals (48.8582222222222, 24.2945)` のシーケンスを返します。
- **`altova:parse-geolocation`**`('48°51''29.6"N  24°17''40.2"')` は 2 つの `xs:decimals (48.8582222222222, 24.2945)` のシーケンスを返します。
- **`altova:parse-geolocation`**`( image-exif-data(//MyImages/Image20141130.01)/@Geolocation )` は 2 つの `xs:decimals` のシーケンスを返します。

位置情報入力文字列フォーマット :

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が 1 つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は `+90` から `-90` (北 から南) です。経度の値の範囲は `+180` から `180` (東 から西) です。

メモ: 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い (`''`) でハイライトされており、エスケープした単位インジケーターは青い (`""`) でハイライトされています。

- 度、分、10進の秒、方角のサフィックス付き (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'66.66"W**

- 度、分、10進の秒、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'66.66"**

- 度、分、10進の分、方角のサフィックス付き (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10進の分、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, W/E)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:
33.33N -22°44'66.66"
33.33 22°44'66.66"W
33.33 22.c

- *Altova Exif 属性 :位置情報*

  Altova XPath/XQuery エンジンはカスタム属性 **Geolocation** を標準 Exif メタデータタグから生成します。**Geolocation** は、4 つの Exif タグの連結です :単位の追加された (下のテーブル参照) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33° 51'21.91"S 151° 13'11.73"E

- **geolocation-distance-km [altova:]**

  altova:geolocation-distance-km(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) を **xs:decimal** とする **XP3 XQ3**
  2 つの位置情報の間の距離をキロメートルで計算します。位置情報入力文字列を提供することのできるフォーマットは下にリストされています。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から -180 (東 から西) です。

  メモ: image-exif-data 関数と Exif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24° 17'40.2""") は xs:decimal 4183.08132372392 を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から 180 (東 から西) です。

メモ: 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、10進の秒、方角のサフィックス付き (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'66.66"W**

- 度、分、10進の秒、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'66.66"**

- 度、分、10進の分、方角のサフィックス付き (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10進の分、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, W/E)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:
33.33N -22°44'66.66"
33.33 22°44'66.66"W
33.33 22.c

Altova Exif 属性 :位置情報

Altova XPath/XQuery エンジンはカスタム属性 **Geolocation** を標準 Exif メタデータタグから生成します。**Geolocation** は、4つの Exif タグの連結です :単位の追加された (下のテーブル参照) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°

				51'21.91"S 151° 13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) を**xs:decimal** とする XP3 XQ3

2 つの位置情報の間の距離をマイルで計算します。位置情報入力文字列を提供することのできるフォーマットは下にリストされています。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から 180 (東 から西) です。

メモ: image-exif-data 関数とExif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。.

サンプル

- **altova:geolocation-distance-mi**("33.33 -22.22", "48°51'29.6""N 24° 17'40.2""") は xs:decimal 2599.40652340653 を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から 180 (東 から西) です。

メモ: 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用符が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、 10進の秒、方角のサフィックス付き (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'66.66"W**

- 度、分、 10進の秒、プレフィックスサイン付き (+/-); (N/W) のためのプラスサインは任意です。
  +/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'66.66"**

- 度、分、 10進の分、方角のサフィックス付き (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、 10進の分、プレフィックスサイン付き (+/-); (N/W) のためのプラスサインは任意です。
  +/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, W/E)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (N/W) のためのプラスサインは任意です。
  +/-D.DD +/-D.DD
  サンプル: 33.33 -22.22

フォーマットの組み合わせのサンプル:
33.33N -22°44'66.66"
33.33 22°44'66.66"W
33.33 22.c

Altova Exif 属性 :位置情報

Altova XPath/XQuery エンジンはカスタム属性 Geolocation を標準 Exif メタデータタグから生成します。Geolocation は、4 つの Exif タグの連結です :単位の追加された (下のテーブル参照) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33° 51'21.91"S 151° 13'11.73"E

geolocation-within-polygon [altova:]

altova:geolocation-within-polygon(Geolocation as xs:string, ((PolygonPoint as xs:string)+)) を xs:boolean とする XP3 XQ3

PolygonPoint 引数により説明されている Geolocation (最初の引数) が多角形のエリア内に存在するかを決定します。もし、PolygonPoint 引数が (最初と最後のポイントが同じ場合に作成される) 閉じられたフィギュアを作成しない場合、フィギュアを閉じるために、最初のポイントが明示的に最後のポイントとして追加されます。全ての引数 (Geolocation および PolygonPoint+) は、(下にリストされるフォーマットの) 位置情報入力文字列により提供されます。Geolocation 引数が多角形エリア内にある場合、関数は true(); を返します。その他の場合は false() を返します。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から -180 (東 から西) です。

メモ: image-exif-data 関数と Exif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32")) は true() を返します。
- altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24")) は true() を返します。
- altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48°51'29.6""N 24°17'40.2""")) は true() を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が 1 つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (北 から南) です。経度の値の範囲は +180 から 180 (東 から西) です。

メモ: 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(**''**) でハイライトされており、エスケープした単位インジケーターは青い(**""**) でハイライトされています。

- 度、分、10進の秒、方角のサフィックス付き (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'66.66"W**

- 度、分、10進の秒、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'66.66"**

- 度、分、10進の分、方角のサフィックス付き (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10進の分、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, W/E)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
  +/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:
33.33N -22°44'66.66"
33.33 22°44'66.66"W
33.33 22.c

Altova Exif 属性 :位置情報

Altova XPath/XQuery エンジンはカスタム属性 **Geolocation** を標準 Exif メタデータタグから生成します。**Geolocation** は、4 つの Exif タグの連結です :単位の追加された (*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33° 51'21.91"S 151° 13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geolocation *as xs:string*, **RectCorner-1** *as xs:string*, **RectCorner-2** *as xs:string*) を **xs:boolean** とする **XP3 XQ3**

長方形の対格の角を指定する第 2及び第 3引数 (**RectCorner-1**、**RectCorner-2**)により説明されている**Geolocation** (最初の引数)が長方形のエリア内に存在するかを決定します。全ての引数 (Geolocation、**RectCorner-1** および**RectCorner-2**) は、 (*下にリストされるフォーマットの*) 位置情報入力文字列により提供されます。Geolocation 引数が長方形エリア内にある場合、関数はtrue(); を返します。その他の場合はfalse()を返します。緯度の値の範囲は+90 から-90 (北 から南)です。経度の値の範囲は+180 から 180 (東 から西)です。

メモ:image-exif-data 関数とExif メタデータの@Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

- *サンプル*
  - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", "-48 24"**)** は true() を返します。
  - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", "48 24"**)** は false() を返します。
  - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", **"**48°51'29.6**""**S 24°17'40.2**""")** はtrue() を返します。

- *位置情報入力文字列フォーマット:*

  位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番)を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は+90 から-90 (北 から南)です。経度の値の範囲は+180 から 180 (東 から西)です。

  メモ:単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(**"**) でハイライトされており、エスケープした単位インジケーターは青い(**""**) でハイライトされています。

  - 度、分、 10進の秒、方角のサフィックス付き (N/S, W/E)
    D°M'S.SS"N/S D°M'S.SS"W/E
    サンプル: **33°55'11.11"N 22°44'66.66"W**

  - 度、分、 10進の秒、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
    +/-D°M'S.SS" +/-D°M'S.SS"
    サンプル: **33°55'11.11" -22°44'66.66"**

  - 度、分、 10進の分、方角のサフィックス付き (N/S, W/E)
    D°M.MM'N/S D°M.MM'W/E
    サンプル: **33°55.55'N 22°44.44'W**

  - 度、分、 10進の分、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
    +/-D°M.MM' +/-D°M.MM'
    サンプル: **+33°55.55' -22°44.44'**

  - 10 進の度、方角のサフィックス付き (N/S, W/E)
    D.DDN/S D.DDW/E
    サンプル: **33.33N 22.22W**

  - 10 進の度、プレフィックスサイン付き (+/-); (**N/W**) のためのプラスサインは任意です。
    +/-D.DD +/-D.DD

サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:
33.33N -22°44'66.66"
33.33 22°44'66.66"W
33.33 22.c

- *Altova Exif 属性 :位置情報*

  Altova XPath/XQuery エンジンはカスタム属性 **Geolocation**を標準 Exif メタデータタグから生成します。**Geolocation** は、4つのExif タグの連結です :単位の追加された (下のテーブル参照) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33° 51'21.91"S 151° 13'11.73"E

[Top]

XPath/ XQuery 関数 :イメージに関連

以下のイメージに関連したXPath/XQuery 拡張関数は、MapForce の現在のバージョンによりサポートされています。また、次で使用することができます : (i) XSLT コンテキスト内のXPath 式、または (ii) XQuery ドキュメント内のXQuery 式。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3**

suggested-image-file-extension [altova:]

altova:suggested-image-file-extension(Base64String *as string***)** を**string?** とする **XP3** **XQ3**

イメージファイルのBase64 エンコードを引数として、イメージのファイル拡張子を、イメージのBase64エンコード内の記録として返します。返された値は、エンコード内で使用することのできるイメージ型情報を基にしたヒントです。この情報が使用できない場合、空の文字列が返されます。この関数は、Base64 イメージをファイルとして保存し、適切なファイル拡張子を動的取得する際に役に立ちます。

- *サンプル*

- **altova:suggested-image-file-extension**(/MyImages/MobilePhone/Image20141130.01) は 'jpg' を返します。
- **altova:suggested-image-file-extension**($XML1/Staff/Person/@photo) は '' を返します。

上のサンプルでは、関数の引数として与えられたノードはBase64 エンコードイメージを含むと仮定します。最初のサンプルは jpg をファイルの型および拡張子として取得します。二番目のサンプルでは、与えられたBase64 エンコードは使用できる拡張子の情報を提供しません。

mt-transform-image [altova:]

altova:mt-transform-image(Base64Image *as Base64BinaryString*, **Size** *as item()*+, **Rotation** *as xs:integer*, **Quality** *as xs:integer*) を**Base64BinaryString** とする **XP3** **XQ3**

Base64-エンコードイメージを最初の引数として、変換されたBase64-エンコードイメージを返します。第 2、第 3、第 4引数は変換された以下のイメージパラメーターです :サイズ、回転、およびクオリティ。

- サイズ引数には 3つのサイズ変更のオプションがあります。

(X,Y)	絶対ピクセルの値。アスペクト率は保持されません。高さと幅は自動的にイメージの長いおよび短い辺に逢わされるため、高さと幅の指示は関係ありません。 2つの整数アイテムのシーケンスとして値が入力されます。かっこが必要です。
X	X をピクセルの新しい長い辺として、イメージの縦横比のサイズ変更が行われます。アスペクト率は保持されます。値は整数で引用符なしで入力されます。
'X%'	元のディメンションの与えられたパーセンテージにイメージのサイズを変更します。値は文字列として引用符を付けて入力される必要があります。

- 回転は以下の値を持つことができます :90、180、270、-90、-180、-270。これらの値は回転の度数です。正の値はイメージを時計回りに回転させます。負の値はイメージを反時計回りに回転させます。Altova Exif 属性 **OrientationDegree** を使用して、イメージの現在の回転の度数 (0, 90, 180, 270) をイメージのExif **Orientation** タグから取得することができます。ですが、**OrientationDegree** 属性はデータの**Orientation** タグから取得されるため、Exif データ内に **Orientation** タグが存在する場合のみ使用することができます。 (*下のOrientationDegree 説明を参照してください*)。
- クオリティは、0 から100 の値で、JPEG 圧縮のIJG クオリティスケールの値を参照していますが、クオリティのパーセンテージのインジケーターではありません。サイズとクオリティのどちらかを優先すると、もう一方の優先度が下がります。フルカラーソースのイメージでは、75 が通常最高値と見なされています。もし、75 が満足のいく結果をもたらさない場合は、値を上げてください。

メモ: Exif データが元のイメージに存在する場合、変換時に削除され、変換されたイメージにはExif データは存在しません。

サンプル

- **mt-transform-image**(Images/Image[@id='43'], '50%', 90, 75)
  関数は、43 の@id 値を持つイメージ子孫ノード内でBase64-エンコード文字列として保管されているイメージを入力とします。関数は変換されたイメージを返します。変換されたイメージは 50% まで、サイズ変更され、時計回りに 90度回転され、 75のクオリティレベルを与えられます。
- **mt-transform-image**(Images/Image[@id='43'], 400, 90, 75)
  関数は前のサンプルと同じ結果を出しますが、長い辺は400 ピクセルの特定の値に設定されています。元のイメージのアスペクト率は保持されます。
- **mt-transform-image**(Images/Image[@id='43'], (400, 280), image-exif-

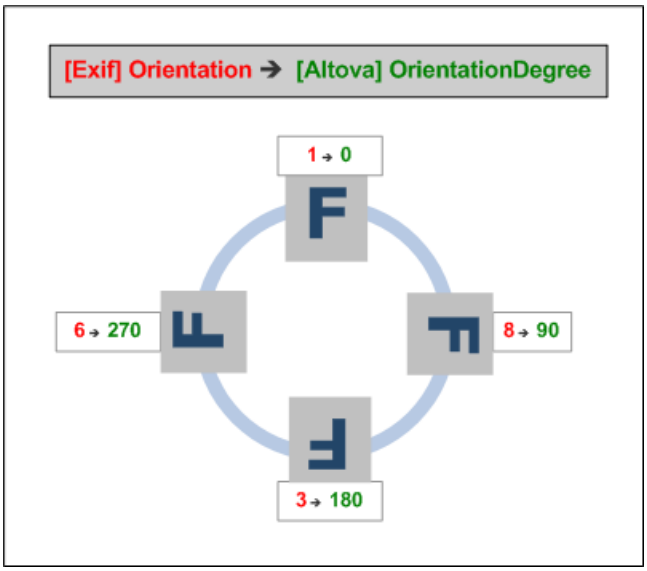
```
data($XML1/$XML1/Images/ReferenceImage)/@OrientationDegree, 75)
```

このサンプルは、前のサンプルと同じイメージを選択し、同じクオリティ値 (75) を設定します。イメージのサイズは 400x280 ピクセルに設定されています。回転値は、ノード内の Base64-エンコードイメージの **@OrientationDegree** 属性から得られます。

Altova Exif 属性 :OrientationDegree

Altova XPath/XQuery エンジンはカスタム属性 **OrientationDegree** を Exif メタデータタグ **Orientation** から生成します。

OrientationDegree は標準 Exif タグ Orientation を正数の値 (1、8、3 または 6) からそれぞれ対応する度数の値 (0, 90, 180, 270) へ下の図に示されているように変換します。2、4、5、7 の Orientation 値の変換はないことに注意してください。これらの向きはイメージ 1 を垂直方向中央軸で反転して、2 の値を持つイメージを取得します。そして、このイメージを 90 度ごとに時計回りにジャンプさせそれぞれ 7、4、および 5 の値を取得します。)。



標準 Exif メタタグ

- ImageWidth
- ImageLength
- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation
- SamplesPerPixel
- PlanarConfiguration
- YCbCrSubSampling
- YCbCrPositioning
- XResolution
- YResolution
- ResolutionUnit
- StripOffsets
- RowsPerStrip
- StripByteCounts

- JPEGInterchangeFormat
- JPEGInterchangeFormatLength
- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod

(C) 2017 Altova GmbH

- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

▾ image-exif-data [altova:]

altova:image-exif-data(Base64BinaryString *as string*) を **element?** とする **XP3 XQ3**

Base64 エンコードイメージを引数として、イメージの Exif メタデータを含む **Exif** という名の要素を返します。The

Exif メタデータは Exif 要素の属性の値ペアとして作成されます。属性名は、Base64エンコード内で検出された Exif データタグです。Exif 仕様タグのリストは以下の通りです。ベンダー特有のタグが Exif データ内に存在する場合、タグとその値も属性値のペアとして返されます。標準 Exif メタデータタグに追加して (*下のリスト参照*) Altova 特有の属性値のペアもまた生成されます。これらの Altova Exif 属性は以下のとおりです。

サンプル

- 属性にアクセスするには、以下の関数を使用します :
  image-exif-data(//MyImages/Image20141130.01)/**@GPSLatitude**
  image-exif-data(//MyImages/Image20141130.01)/**@Geolocation**
- 全ての属性にアクセスするには、以下の関数を使用します :
  image-exif-data(//MyImages/Image20141130.01)/**@***
- 全ての属性の名前にアクセスするには、以下の式を使用します :
  for $i **in** **image-exif-data**(//MyImages/Image20141130.01)/**@*** **return** **name**($i)
  関数により返される属性の名前を検出するために役に立ちます。

Altova Exif 属性 :位置情報

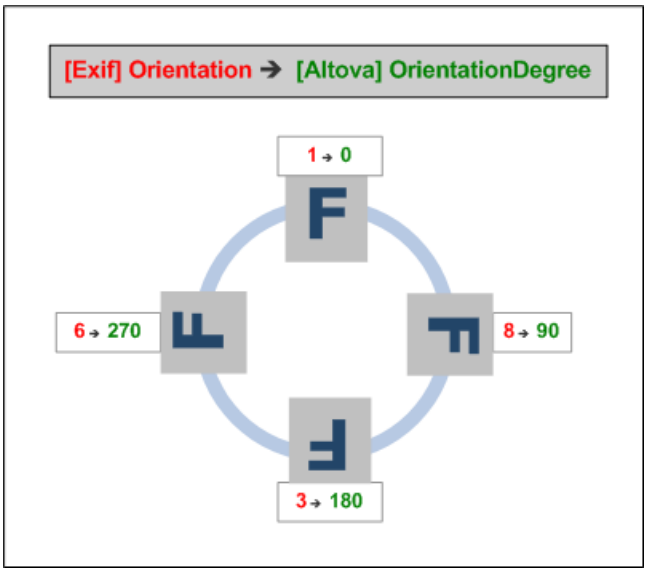
Altova XPath/XQuery エンジンはカスタム属性 **Geolocation** を標準 Exif メタデータタグから生成します。**Geolocation** は、4つの Exif タグの連結です :単位の追加された (*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33° 51'21.91"S 151° 13'11.73"E

Altova Exif 属性 :OrientationDegree

Altova XPath/XQuery エンジンはカスタム属性 **OrientationDegree** を Exif メタデータタグ **Orientation** から生成します。

OrientationDegree は標準 Exif タグ Orientation を正数の値 (1、8、3 または 6) からそれぞれ対応する度数の値 (0, 90, 180, 270) へ下の図に示されているように変換します。2、4、5、7 の Orientation 値の変換はないことに注意してください。(これらの向きはイメージ 1 を垂直方向中央軸で反転して、2 の値を持つイメージを取得します。そして、このイメージを 90 度ごとに時計回りにジャンプさせそれぞれ 7、4、および 5 の値を取得します。)



標準 Exif メタデータタグのリスト

- ImageWidth
- ImageLength
- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation
- SamplesPerPixel
- PlanarConfiguration
- YCbCrSubSampling
- YCbCrPositioning
- XResolution
- YResolution
- ResolutionUnit
- StripOffsets
- RowsPerStrip
- StripByteCounts
- JPEGInterchangeFormat
- JPEGInterchangeFormatLength
- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion

- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

(C) 2017 Altova GmbH

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[トップ]

XPath/ XQuery 関数 :数値

Altova の数値拡張関数は XPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、**`http://www.altova.com/xslt-extensions` に収められており、`altova:` プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意ください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	`XP1` `XP2` `XP3`
XSLT 関数 (XSLT 内のXPath 式で使用):	`XSLT1` `XSLT2`

	XSLT3
XQuery 関数 (XQuery 内のXQuery 式で使用):	XQ1 XQ3

自動付番関数

generate-auto-number [altova:]

altova:generate-auto-number(ID *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) を xs:integer とする XP1 XP2 XQ1 XP3 XQ3

関数が呼び出される度に番号を生成します。関数が初めて呼び出される際に生成される最初の番号は StartsWith 引数により指定されます。その後の関数の呼び出しは新しい番号を生成します。この番号は前に生成された番号を、Increment 引数で指定された値ごとにインクリメントします。実質的には、altova:generate-auto-number 関数は、ID 引数により指定されたカウンターを作成し、このカウンターが関数が呼び出されるごとにインクリメントされます。ResetOnChange 引数の値が、前の関数呼び出しから変更されると、生成される番号の値が StartsWith 値にリセットされます。自動付番は altova:reset-auto-number 関数を使用してリセットすることができます。

サンプル

- **altova:generate-auto-number**("ChapterNumber", 1, 1, "SomeString") は、関数が呼び出される度に番号を 1 つ返します。starting with 1, から始まり、関数が呼び出される度に 1 ずつインクリメントします。その後の呼び出しの 4 番目の"SomeString"引数がある限り、インクリメントは継続されます。4 番目の引数の値が変更されると、(ChapterNumber と呼ばれる)カウンターは 1 にリセットされます。ChapterNumber の値も altova:reset-auto-number 関数の呼び出しにより次の altova:reset-auto-number("ChapterNumber")ようにリセットされます。

reset-auto-number [altova:]

altova:reset-auto-number(ID *as xs:string*) XP1 XP2 XQ1 XP3 XQ3

この関数は、ID 引数内で名づけられた自動付番カウンターの番号をリセットします。引数内のカウンター名を作成した関数の引数により指定された数値にリセットされます。altova:generate-auto-number

サンプル

- **altova:reset-auto-number**("ChapterNumber") は、altova:generate-auto-number 関数により作成された ChapterNumber という名の自動付番カウンターをリセットします。
- ChapterNumber を作成した altova:generate-auto-number 関数の StartsWith 引数の値に数値をリセットします。

[Top]

数値関数

hex-string-to-integer [altova:]

altova:hex-string-to-integer(HexString *as xs:string*) を xs:integer とする XP3 XQ3

10 進のシステム(10進)内の正数の16進の同値の文字列引数を必要とし、10進の整数を返します。

サンプル

- **altova:hex-string-to-integer**('1') は 1 を返します。
- **altova:hex-string-to-integer**('9') は 9 を返します。

- **altova:hex-string-to-integer**('A') は10 を返します。
- **altova:hex-string-to-integer**('B') は11 を返します。
- **altova:hex-string-to-integer**('F') は15 を返します。
- **altova:hex-string-to-integer**('G') はエラーを返します。
- **altova:hex-string-to-integer**('10') は16 を返します。
- **altova:hex-string-to-integer**('01') は1 を返します。
- **altova:hex-string-to-integer**('20') は32 を返します。
- **altova:hex-string-to-integer**('21') は33 を返します。
- **altova:hex-string-to-integer**('5A') は90 を返します。
- **altova:hex-string-to-integer**('USA') はエラーを返します。

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Integer *as xs:integer*) を**xs:string** とする XP3 XQ3

正数を引数として必要とし、文字列として自身のベース16の同値を返します。

サンプル

- **altova:integer-to-hex-string**(1) は'1' を返します。
- **altova:integer-to-hex-string**(9) は'9' を返します。
- **altova:integer-to-hex-string**(10) は'A' を返します。
- **altova:integer-to-hex-string**(11) は'B' を返します。
- **altova:integer-to-hex-string**(15) は'F' を返します。
- **altova:integer-to-hex-string**(16) は'10' を返します。
- **altova:integer-to-hex-string**(32) は'20' を返します。
- **altova:integer-to-hex-string**(33) は'21' を返します。
- **altova:integer-to-hex-string**(90) は'5A' を返します。

[トップ]

数値をフォーマットする関数

generate-auto-number [altova:]

altova:generate-auto-number(ID *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) を xs:integer とする XP1 XP2 XQ1 XP3 XQ3

関数が呼び出される度に番号を生成します。関数が初めて呼び出される際に生成される最初の番号は StartsWith 引数により指定されます。その後の関数の呼び出しは新しい番号を生成します。この番号は前に生成された番号を、Increment 引数で指定された値ごとにインクリメントします。実質的には、altova:generate-auto-number 関数は、ID 引数により指定されたカウンターを作成し、このカウンターが関数が呼び出されるごとにインクリメントされます。ResetOnChange 引数の値が、前の関数呼び出しから変更されると、生成される番号の値がの StartsWith 値にリセットされます。自動付番は altova:reset-auto-number 関数を使用してリセットすることができます。

サンプル

- **altova:generate-auto-number**("ChapterNumber", 1, 1, "SomeString") は、関数が呼び出される度に番号を 1 つ返します。starting with 1, から始まり、関数が呼び出される度に 1 ずつインクリメントします。その後の呼び出しの 4 番目の"SomeString"引数がである限り、インクリメントは継続されます。4番目の引数の値が変更されると、(ChapterNumber と呼ばれる)カウンターは 1 にリセットされます。ChapterNumber の値も altova:reset-auto-number 関数の呼び出しによりの

altova:reset-auto-number("ChapterNumber")ようにリセットされます。

[トップ]

XPath/ XQuery 関数 :シーケンス

Altova のシーケンス拡張関数は XPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3**

attributes [altova:]

altova:attributes(AttributeName *as xs:string***)** を**attribute()*** とする **XP3** **XQ3**
入力引数 AttributeName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、attribute:: 軸に対して行われます。これは、コンテキストノードが親要素ノードである必要があることを意味します。

サンプル

- **altova:attributes**("MyAttribute") はMyAttribute()* を返します。

altova:attributes(AttributeName *as xs:string*, **SearchOptions** *as xs:string***)** as **attribute()*** **XP3** **XQ3**
入力引数 AttributeName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、attribute:: 軸に対して行われます。コンテキストノードが親要素ノードである必要があります。第 2引数はオプションのフラグを含みます。使用することのできるフラグは以下の通りです :
r = 正規表現検索を切り替えます ;AttributeName は正規表現検索文字列である必要があります ;
f = このオプションが指定されている場合、AttributeName は完全一致を提供します。それ以外の場合は、AttributeName は属性名に部分的に一致するとその属性を返します。例えば : **f** が指定されていない場合、MyAtt はMyAttribute を返します。
i = 大文字と小文字を一致させる検索に切り替えます。
p = 検索に名前空間プレフィックスを含みます。AttributeName は、名前空間プレフィックスを含みます。例えば : altova:MyAttribute。
フラグは順序に関わりなく書き込むことができます。無効なフラグはエラーを生成します。1つまたは複数のフラグを省略することができます。空の文字列は許可されていますが、引数を1つしか持たない関数と同じ効果を持ちます (前の署名)、ですが、空のシーケンスは第 2の引数としては許可されていません。

サンプル

- **altova:attributes**("MyAttribute", "rfip") はMyAttribute()* を返します。

- **altova:attributes**("MyAttribute", "pri") は MyAttribute()* を返します。
- **altova:attributes**("MyAtt", "rip") は MyAttribute()* を返します。
- **altova:attributes**("MyAttributes", "rfip") は不一致を返します。
- **altova:attributes**("MyAttribute", "") は MyAttribute()* を返します。
- **altova:attributes**("MyAttribute", "Rip") 認識されないフラグエラーを返します。
- **altova:attributes**("MyAttribute",) 見つからない第 2引数を返します。

elements [altova:]

altova:elements(ElementName *as xs:string*) を **element()*** とする XP3 XQ3
入力引数 ElementName で与えられた名前と同じローカル名を持つすべての要素を返します。検索は大文字と小文字を区別して child:: 軸に対して実行されます。コンテキストノードは、検索される要素の親ノードである必要があります。

サンプル

- **altova:elements**("MyElement") は MyElement()* を返します。

altova:elements(ElementName *as xs:string*, **SearchOptions** *as xs:string*) as **element()*** XP3 XQ3
入力引数 ElementName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、child:: 軸に対して行われます。コンテキストノードが親要素ノードである必要があります。第 2引数はオプションのフラグを含みます。使用することのできるフラグは以下の通りです：
r = 正規表現検索を切り替えます；ElementName は正規表現検索文字列である必要があります；
f = このオプションが指定されている場合、ElementName は完全一致を提供します。それ以外の場合は、ElementName は属性名に部分的に一致するとその属性を返します。例えば：**f** が指定されていない場合、MyElem は MyElement を返します。
i = 大文字と小文字を一致させる検索に切り替えます。
p = 検索に名前空間プレフィックスを含みます。ElementName は、名前空間プレフィックスを含みます。例えば：altova:MyElement。
フラグは順序に関わりなく書き込むことができます。無効なフラグはエラーを生成します。1つまたは複数のフラグを省略することができます。空の文字列は許可されていますが、引数を1つしか持たない関数と同じ効果を持ちます (前の署名)、ですが、空のシーケンスは第 2の引数としては許可されていません。

サンプル

- **altova:elements**("MyElement", "rip") は MyElement()* を返します。
- **altova:elements**("MyElement", "pri") は MyElement()* を返します。
- **altova:elements**("MyElement", "") は MyElement()* を返します。
- **altova:attributes**("MyElem", "rip") は MyElement()* を返します。
- **altova:attributes**("MyElements", "rfip") 不一致を課wします。
- **altova:elements**("MyElement", "Rip") 認識されないフラグエラーを返します。
- **altova:elements**("MyElement",) 見つからない第 2引数を返します。

find-first [altova:]

altova:find-first((Sequence *as item()**), **(Condition(Sequence-Item** *as xs:boolean*)) を **item()?** とする XP3 XQ3
この関数は 2つの引数を必要とします。最初の引数は 1つ、または 1つ以上のデータ型のアイテムのシーケンスです。第 2の引数Condition は、(**1** のアリティを持つ) 1つの引数 を必要とし、boolean を返す XPath 関数に対する参照です。Condition で参照された関数の代わりに、**Sequence** の各アイテムが提出されます。(注意：この関数は 1つの引数のみを必要とします。) **Condition** 内の関数に **true()** と評価させる最初の **Sequence** アイテムは、**altova:find-first**、反復の終了の結果として返されます。

サンプル

- `altova:find-first(5 to 10, function($a) {$a mod 2 = 0})` は `xs:integer 6` を返します。

Condition 引数は、**$a** という名のインライン関数を宣言し、定義します。XPath 3.0 インライン関数 **function()** を参照します。**altova: Sequence** 引数内の各アイテムでは、**find-first** がパスされ、代わりに、**$a** を入力値とします。入力値は関数定義 (`$a mod 2 = 0`)内の条件に対してテストされます。この条件を満たす最初の入力値が**altova:find-first** (この場合は**6**) の結果として返されます。

- `altova:find-first((1 to 10), (function($a) {$a+3=7}))` は `xs:integer 4` を返します。

更なるサンプル

ファイル**C:\Temp\Customers.xml** が存在する場合：

- `altova:find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` は `xs:string C:\Temp\Customers.xml` を返します。

ファイル**C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** が存在する場合：

- `altova:find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` は `xs:string http://www.altova.com/index.html` を返します。

ファイル**C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** も存在しない場合：

- `altova:find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` 結果無しを返します。

上のサンプルについての注意点

- XPath 3.0 関数 `doc-available` はURI として使用され、ドキュメントノードが提出されたURI で検出される場合 `true` を返す単一の引数を必要とします。 (ですから、提出されたURI でのドキュメントはXML ドキュメントである必要があります。)
- `doc-available` 関数は、`altova:find-first` の第 2引数である、**Condition** で使用することができます。これは、 1つの引数 (アリティ=1)のみを必要とするからであり、`item()`を入力 (URI として使用される文字列) として、boolean の値を返すからです。
- `doc-available` 関数は、参照されているだけで、呼び出されていない点に注意してください。アタッチされている`#1` サフィックスは関数が 1つのアリティであることを表示するためです。`doc-available#1` の意味は以下のとおりです：アリティ=1 を持つ doc-availabe() 関数を使用し、最初のシーケンスの各アイテムの代わりに単一引数としてパスします。 この結果、 2つの文字列の各自つは、文字列をURI として使用し、URI にドキュメントノードが存在するかテストする**doc-available()**にパスされます。1つが各当する場合、**doc-available()** 関数は`true()` を評価し、シーケンス内のその文字列のインデックス ポジションは、 **altova:find-first** 関数の結果として返されます。doc-available() 関数に関しての注意点：相対パスは、デフォルトで関数がロードされるXML ドキュメントの現在のベースURI に対して相対的に解決されます。

find-first-combination [altova:]

altova:find-first-combination((Seq-01 *as item()**), **(Seq-02** *as item()**), **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) を**item()*** とする **XP3** **XQ3**

この関数は 3つの引数を必要とします：

- 最初の 2 つの引数 **Seq-01** と**Seq-02**, は、1 つまたは 1 つ以上のデータ型のアイテムです。
- 第 3 の引数 **Condition** は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 と**Seq-02** のアイテムは指定された組み合わせ (各シーケンスからの 1 つずつにアイテムで構成されるペア) で、**Condition** 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

Condition 関数に**true()** と評価するように指示する最初のペアは**altova:find-first-combination** の結果として返されます。以下の点に注意してください: (i) **Condition** 関数が提出された引数ペア内で繰り返され、**true()** を評価しない場合、**altova:find-first-combination** は、結果を返しません; (ii) **altova:find-first-combination** の結果が常に (データ型の) アイテムのペアである場合またはアイテムでない場合。

サンプル

- `altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` は `xs:integers (11, 21)` のシーケンスを返します。
- `altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` は `xs:integers (11, 22)` のシーケンを返します。
- `altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34})` は `xs:integers (11, 23)` のシーケンを返します。

find-first-pair [altova:]

altova:find-first-pair((Seq-01 *as item()**)**, **(Seq-02** *as item()**)**, **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean***))** を**item()*** とする **XP3** **XQ3**

この関数は 3 つの引数を必要とします:

- 最初の 2 つの引数 **Seq-01** と**Seq-02**, は、1 つまたは 1 つ以上のデータ型のアイテムです。
- 第 3 の引数 **Condition** は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 と**Seq-02** のアイテムは指定された組み合わせで、**Condition** 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Condition 関数に**true()** と評価するように指示する最初のペアは**altova:find-first-pair** の結果として返されます。以下の点に注意してください: (i) **Condition** 関数が提出された引数ペア内で繰り返され、**true()** を評価しない場合、**altova:find-first-pair** は、結果を返しません; (ii) **altova:find-first-combination** の結果が常に (データ型の) アイテムのペアである場合またはアイテムでない場合。

サンプル

- `altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` は `xs:integers (11, 21)` のシーケンスを返します。
- `altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` は*結果無し*を返します。

上の 2 つのサンプルに表示される通り、ペアの順序は以下の通りです: `(11, 21) (12, 22) (13, 23)...(20, 30)`。(33 を返す指示されたペアがないため) この理由で第 2 のサンプルは*結果無し*を返

します。

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Seq-01 *as item()**)**,* **(Seq-02** *as item()**)**,* **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean***))** を**xs:integer** とする **XP3 XQ3**

この関数は 3 つの引数を必要とします :

- 最初の 2 つの引数 **Seq-01** と**Seq-02**, は、1 つまたは1つ以上のデータ型のアイテムです。
- 第 3の引数 **Condition** は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 と**Seq-02** のアイテムは指定された組み合わせで、**Condition** 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Condition 関数に**true()**を評価させる、最初に指示されたペアのインデックスポジションは、は **altova:find-first-pair-pos** の結果として返されます。関数が提出された引数ペア内で繰り返され、**true()**を一度も評価しない場合、**altova:find-first-pair-pos** *結果無し*が返します。

サンプル

- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) は1 を返します
- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) は*結果無し*を返します。

上の 2 つのサンプルに表示される通り、ペアの順序は以下の通りです :(11, 21) (12, 22) (13, 23)...(20, 30)。最初のサンプルでは、最初のペアは**Condition** 関数に**true()**を評価させ、シーケンス内のインデックスポジションには**1** が返されます。第 2 のサンプルでは、33 を返すペアがないため、*結果無し*が返します。

find-first-pos [altova:]

altova:find-first-pos((Sequence *as item()**)**,* **(Condition(Sequence-Item** *as xs:boolean***))** を**xs:integer** とする **XP3 XQ3**

この関数は 2 つの引数を必要とします。最初の引数は 1 つ、または 1 つ以上のデータ型のアイテムのシーケンスです。第 2 の引数 Condition は、(**1** のアリティを持つ) 1 つの引数 を必要とし、boolean を返す XPath 関数に対する参照です。Condition で参照された関数の代わりに、**Sequence** の各アイテムが提出されます。(*注意 : この関数は 1 つの引数のみを必要とします。*)**Condition** 内の関数に**true()**と評価させる最初の **Sequence** アイテムは、**altova:find-first-pos** の結果として返された**Sequence** 内インデックスポジションを持ちます。

サンプル

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) は xs:integer 2 を返します。

Condition 引数は、**$a** という名のインライン関数を宣言し、定義します。XPath 3.0 インライン関数 **function()**を参照します。 **Sequence** 引数内の各アイテムでは、**find-first-pos** がパスされ、代わりに、**$a** を入力値とします。入力値は関数定義 ($a mod 2 = 0)内の条件に対してテストされます。この条件を満たす最初の入力値が**altova:find-first-pos** ((シーケンス内で条件を満たす最初の値である 6 がシーケンスのインデックス位置 2 にあるためこの場合は 2、)の結果として返されます。

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) は xs:integer 3 を返します。

更なるサンプル
ファイル **C:\Temp\Customers.xml** が存在する場合：

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) returns 1

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** が存在する場合：

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) returns 2

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** も存在しない場合：

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) 結果無しを返します。

上のサンプルについての注意点

- XPath 3.0 関数 doc-available は URI として使用され、ドキュメントノードが提出された URI で検出される場合 true を返す単一の引数を必要とします。(ですから、提出された URI でのドキュメントは XML ドキュメントである必要があります。)
- doc-available 関数は、altova:find-first-pos の第 2 引数である、**Condition** で使用することができます。これは、1 つの引数 (アリティ=1) のみを必要とするからであり、item() を入力 (URI として使用される文字列) として、boolean の値を返すからです。
- doc-available 関数は、参照されているだけで、呼び出されていない点に注意してください。アタッチされている #1 サフィックスは関数が 1 つのアリティであることを表示するためです。doc-available#1 の意味は以下のとおりです：アリティ=1 を持つ doc-availabe() 関数を使用し、最初のシーケンスの各アイテムの代わりに単一引数としてパスします。この結果、2 つの文字列の各自つは、文字列を URI として使用し、URI にドキュメントノードが存在するかテストする **doc-available()** にパスされます。1 つが各当する場合、**doc-available()** 関数は true() を評価し、シーケンス内のその文字列のインデックス ポジションは、**altova:find-first-pos** 関数の結果として返されます。doc-available() 関数に関しての注意点：相対パスは、デフォルトで関数がコードされる XML ドキュメントの現在のベース URI に対して相対的に解決されます。

substitute-empty [altova:]

altova:substitute-empty(FirstSequence *as item()**, **SecondSequence** *as item()***)** を **item()*** とする XP3 XQ3

FirstSequence が空の場合、SecondSequence を返します。FirstSequence が空でない場合、FirstSequence を返します。

サンプル

- **altova:substitute-empty**((1,2,3), (4,5,6)) は (1,2,3) を返します。
- **altova:substitute-empty**((), (4,5,6)) は (4,5,6) を返します。

XPath/ XQuery 関数 :文字列

Altova の文字列拡張関数は XPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、**`http://www.altova.com/xslt-extensions` に収められており、`altova:` プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意ください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	`XP1` `XP2` `XP3`
XSLT 関数 (XSLT 内のXPath 式で使用):	`XSLT1` `XSLT2` `XSLT3`
XQuery 関数 (XQuery 内のXQuery 式で使用):	`XQ1` `XQ3`

camel-case [altova:]

altova:camel-case(InputString *as xs:string***)** を**xs:string** とする `XP3` `XQ3`
入力文字列を**InputString** をキャメルケースで返します。文字列は (空白スペースのショートカットである) 正規表現 `'\s'` を使用して分析されます。空白文字または連続する空白文字のシーケンスの後の最初の非空白スペース文字は大文字です。出力文字列の最初の文字は大文字です。

サンプル

- **altova:camel-case**("max") Max を返します。
- **altova:camel-case**("max max") Max Max を返します。
- **altova:camel-case**("file01.xml") File01.xml を返します。
- **altova:camel-case**("file01.xml file02.xml") File01.xml File02.xml を返します。
- **altova:camel-case**("file01.xml file02.xml") File01.xml File02.xml を返します。
- **altova:camel-case**("file01.xml -file02.xml") File01.xml -file02.xml を返します。

altova:camel-case(InputString *as xs:string*, **SplitChars** *as xs:string*, **IsRegex** *as xs:boolean***)** を**xs:string** とする `XP3` `XQ3`
SplitChars を使用して、次の大文字をトリガーする文字を決定し、入力文字列を**InputString** キャメルケースに変換します。**SplitChars** は、**IsRegex = true()**の場合、または、**IsRegex = false()**の場合、プレーン文字は正規表現として使用されます。出力文字列の最初の文字は大文字です。

サンプル

- **altova:camel-case**("setname getname", "set|get", true()) setName getName を返します。
- **altova:camel-case**("altova\documents\testcases", "\", false()) Altova \Documents\Testcases を返します。

char [altova:]

altova:char(Position *as xs:integer***)** を**xs:string** とする **XP3 XQ3**
xs:string.Position に対してのコンテキストアイテムの値を変換することにより得られた文字列内の Position 引数により指定されたポジションにある文字を含む文字列を返します。引数により提出されたインデックスに文字が存在しない場合、結果文字列は空です。

サンプル

コンテキストアイテムが**1234ABCD** の場合：

- **altova:char**(2) は2 を返します。
- **altova:char**(5) はA を返します。
- **altova:char**(9) は空の文字列を返します。
- **altova:char**(-2) は空の文字列を返します。

altova:char(InputString *as xs:string*, **Position** *as xs:integer***)** を**xs:string** とする **XP3 XQ3**
引数として提出された文字列内のPosition 引数により指定されたポジションでの文字を含む文字列を返します。Position 引数により提出されたインデックスに文字が存在しない場合、結果文字列は空です。

サンプル

- **altova:char**("2014-01-15", 5) は- を返します。
- **altova:char**("USA", 1) はU を返します。
- **altova:char**("USA", 10) は空の文字列を返します。
- **altova:char**("USA", -2) は空の文字列を返します。

first-chars [altova:]

altova:first-chars(X-Number *as xs:integer***)** を**xs:string** とする **XP3 XQ3**
xs:string に対するコンテキストアイテムの値を変換することにより得られた最初のX-Number 文字を含む文字列を返します。

サンプル

コンテキストアイテムが**1234ABCD** の場合：

- **altova:first-chars**(2) は12 を返します。
- **altova:first-chars**(5) は1234A を返します。
- **altova:first-chars**(9) は1234ABCD を返します。

altova:first-chars(InputString *as xs:string*, **X-Number** *as xs:integer***)** を**xs:string** とする **XP3 XQ3**
InputString 引数として提出された文字列の最初の文字を含む文字列を返します。

サンプル

- **altova:first-chars**("2014-01-15", 5) は2014- を返します。
- **altova:first-chars**("USA", 1) はU を返します。

last-chars [altova:]

altova:last-chars(X-Number *as xs:integer***)** を**xs:string** とする **XP3 XQ3**
xs:string に対してのコンテキストアイテムの値の変換より取得された文字列の最後のX-Number 文字を含んでいる文字列を返します。

サンプル

コンテキストアイテムが**1234ABCD** の場合：

- **altova:last-chars**(2) はCD を返します。
- **altova:last-chars**(5) は4ABCD を返します。

- **altova:last-chars**(9) は1234ABCD を返します。

altova:last-chars(InputString *as xs:string*, **X-Number** *as xs:integer*) as **xs:string** とする XP3 XQ3
引数として提出された文字列の最後のX-Number 文字を含んでいる文字列を返します。

サンプル

- **altova:last-chars**("2014-01-15", 5) は01-15 を返します。
- **altova:last-chars**("USA", 10) はUSA を返します。

pad-string-left [altova:]

altova:pad-string-left(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) を**xs:string** とする XP3 XQ3
PadCharacter 引数は1文字です。文字列の左側にパッドされ、この数が、StringLength 引数の整数の値と等しくなるようにStringToPad の文字の数を増やします。StringLength 引数は任意の整数の値（正数または負数）を持つことができますが、パディングは、StringLength の値がStringToPad 内の文字数よりも多い場合の場合のみ発生します。もし、StringToPad がStringLength の値よりも多くの文字数も持つ場合、StringToPad は変更されません。

サンプル

- **altova:pad-string-left**('AP', 1, 'Z') は'AP' を返します。
- **altova:pad-string-left**('AP', 2, 'Z') は'AP' を返します。
- **altova:pad-string-left**('AP', 3, 'Z') は'ZAP' を返します。
- **altova:pad-string-left**('AP', 4, 'Z') は'ZZAP' を返します。
- **altova:pad-string-left**('AP', -3, 'Z') は'AP' を返します。
- **altova:pad-string-left**('AP', 3, 'YZ')は[パッド文字が長すぎます] エラーを返します。

pad-string-right [altova:]

altova:pad-string-right(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) を**xs:string** とする XP3 XQ3
PadCharacter 引数は1文字です。文字列の右側にパッドされ、この数が、StringLength 引数の整数の値と等しくなるようにStringToPad の文字の数を増やします。StringLength 引数は任意の整数の値（正数または負数）を持つことができますが、パディングは、StringLength の値がStringToPad 内の文字数よりも多い場合の場合のみ発生します。もし、StringToPad がStringLength の値よりも多くの文字数も持つ場合、StringToPad は変更されません。

サンプル

- **altova:pad-string-right**('AP', 1, 'Z') を'AP' を返します。
- **altova:pad-string-right**('AP', 2, 'Z') を'AP' を返します。
- **altova:pad-string-right**('AP', 3, 'Z') を'APZ' を返します。
- **altova:pad-string-right**('AP', 4, 'Z') を'APZZ' を返します。
- **altova:pad-string-right**('AP', -3, 'Z') を'AP' を返します。
- **altova:pad-string-right**('AP', 3, 'YZ') は[パッド文字が長すぎます] エラーを返します。

repeat-string [altova:]

altova:repeat-string(InputString *as xs:string*, **Repeats** *as xs:integer*) を**xs:string** とする XP2 XQ1 XP3 XQ3
最初のInputString引数により構成される文字列、Repeats 回繰り返してを生成します。

サンプル

- `altova:repeat-string("Altova #", 3)` は `"Altova #Altova #Altova #"` を返します。

substring-after-last [altova:]

altova:substring-after-last(MainString *as xs:string*, **CheckString** *as xs:string*) を**xs:string** とする **XP3** **XQ3**

CheckString が MainString 内で検出された場合、MainString 内の CheckString が発生した後のサブ文字列が返されます。MainString 内で CheckString が検出されない場合、空の文字列が返されます。CheckString が空の文字列の場合、MainString 全体が返されます。一度以上発生する場合、CheckString の最後の発生の後のサブ文字列が返されます。

サンプル

- `altova:substring-after-last('ABCDEFGH', 'B')` は `'CDEFGH'` を返します。
- `altova:substring-after-last('ABCDEFGH', 'BC')` は `'DEFGH'` を返します。
- `altova:substring-after-last('ABCDEFGH', 'BD')` は `''` を返します。
- `altova:substring-after-last('ABCDEFGH', 'Z')` は `''` を返します。
- `altova:substring-after-last('ABCDEFGH', '')` は `'ABCDEFGH'` を返します。
- `altova:substring-after-last('ABCD-ABCD', 'B')` は `'CD'` を返します。
- `altova:substring-after-last('ABCD-ABCD-ABCD', 'BCD')` は `''` を返します。

substring-before-last [altova:]

altova:substring-before-last(MainString *as xs:string*, **CheckString** *as xs:string*) を**xs:string** とする **XP3** **XQ3**

CheckString が MainString 内で検出された場合、MainString 内の CheckString が発生する前のサブ文字列が返されます。MainString 内で CheckString が一度以上発生する場合、CheckString の最後の発生の前のサブ文字列が返されます。

サンプル

- `altova:substring-before-last('ABCDEFGH', 'B')` は `'A'` を返します。
- `altova:substring-before-last('ABCDEFGH', 'BC')` は `'A'` を返します。
- `altova:substring-before-last('ABCDEFGH', 'BD')` は `''` を返します。
- `altova:substring-before-last('ABCDEFGH', 'Z')` は `''` を返します。
- `altova:substring-before-last('ABCDEFGH', '')` は `''` を返します。
- `altova:substring-before-last('ABCD-ABCD', 'B')` は `'ABCD-A'` を返します。
- `altova:substring-before-last('ABCD-ABCD-ABCD', 'ABCD')` は `'ABCD-ABCD-'` を返します。

substring-pos [altova:]

altova:substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*) を**xs:integer** とする **XP3** **XQ3**

StringToCheck 内での StringToFind の最初の発生の文字位置を整数として返します。StringToCheck の最初の文字は、位置 1 にあります。StringToFind が StringToCheck 内で発生しない場合、整数 0 が返されます。第 2 またはその後の StringToCheck, 発生を確認するには、この関数の次の署名を確認してください。

サンプル

- `altova:substring-pos('Altova', 'to')` 3 を返します。

- **altova:substring-pos**('Altova', 'tov') 3 を返します。
- **altova:substring-pos**('Altova', 'tv') returns 0 を返します。
- **altova:substring-pos**('AltovaAltova', 'to') 3 を返します。

altova:substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*, **Integer** *as xs:integer*) を**xs:integer** とする XP3 XQ3
StringToCheck 内でのStringToFind の最初の発生の文字位置を返します。Integer 引数により与えられた文字位置からStringToFind の検索が開始されます。この位置の前の文字サブ文字列は検索されません。しかし、返された整数は、全体文字列 StringToCheck の検索された文字列の位置です。この署名は、StringToCheck 内で複数回発生する、第 2 または後の発生を検索する際に役に立ちます。StringToFind がStringToCheck 内で発生しない場合、整数 0 が返されます。

サンプル

- **altova:substring-pos**('Altova', 'to', 1) 3 を返します。
- **altova:substring-pos**('Altova', 'to', 3) 3 を返します。
- **altova:substring-pos**('Altova', 'to', 4) 0 を返します。
- **altova:substring-pos**('Altova-Altova', 'to', 0) 3 を返します。
- **altova:substring-pos**('Altova-Altova', 'to', 4) 10 を返します。

trim-string [altova:]

altova:trim-string(InputString *as xs:string*) を**xs:string** とする XP3 XQ3
この関数はxs:string 引数を必要とし、先頭または後続の空白を削除し、トリミングされたxs:string を返します。

サンプル

- **altova:trim-string**(" Hello World ")) は"Hello World" を返します。
- **altova:trim-string**("Hello World ")) は"Hello World" を返します。
- **altova:trim-string**(" Hello World")) は"Hello World" を返します。
- **altova:trim-string**("Hello World")) は"Hello World" を返します。
- **altova:trim-string**("Hello World"))は "Hello World" を返します。

trim-string-left [altova:]

altova:trim-string-left(InputString *as xs:string*) を**xs:string** とする XP3 XQ3
この関数はxs:string 引数を必要とし、先頭または後続の空白を削除し、トリミングされたxs:string を返します。

サンプル

- **altova:trim-string-left**(" Hello World "))は "Hello World " を返します。
- **altova:trim-string-left**("Hello World ")) は"Hello World " を返します。
- **altova:trim-string-left**(" Hello World")) は"Hello World" を返します。
- **altova:trim-string-left**("Hello World")) は"Hello World" を返します。
- **altova:trim-string-left**("Hello World")) は"Hello World" を返します。

trim-string-right [altova:]

altova:trim-string-right(InputString *as xs:string*) を**xs:string** とする XP3 XQ3
この関数はxs:string 引数を必要とし、先頭または後続の空白を削除し、トリミングされたxs:string を返します。

サンプル

- **altova:trim-string-right**(" Hello World "))は" Hello World" を返します。
- **altova:trim-string-right**("Hello World "))は"Hello World" を返します。
- **altova:trim-string-right**(" Hello World")) は" Hello World" を返します。
- **altova:trim-string-right**("Hello World")) は"Hello World" を返します。
- **altova:trim-string-right**("Hello World")) は"Hello World" を返します。

XPath/ XQuery 関数 :その他

以下の一般使用のためのXPath/XQuery 拡張関数は、MapForce の現在のバージョンによりサポートされています。また、次で使用することができます : (i) XSLT コンテキスト内のXPath 式、または (i) XQuery ドキュメント内のXQuery 式。

関数の名前指定と言語の適用性

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間、http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3**

URI 関数

get-temp-folder [altova:]

altova:get-temp-folder() をxs:string とする **XP2** **XQ1** **XP3** **XQ3**

この関数は引数を必要としません。この関数は現在のユーザーの一時的なフォルダーへのパスを返します。

サンプル

- **altova:get-temp-folder**() は、マシン上で xs:string としてC:\Users\<UserName>\AppData\Local\Temp\ と類似したパスを返します。

[Top]

チャート関数

以下に示されるチャート関数により、チャートをイメージとして作成、生成、保存することができます。これらの機能は、お使いのAltova 製品で、以下に記される方法によりサポートされます。しかしながら、将来使用されるバージョンの製品において、以下にある関数のサポートが打ち切られたり、個々の振る舞いが変化する可能性があることに留意してください。将来のリリースにおけるAltova 拡張関数のサポートについては、そのバージョンのドキュメンテーションを参照ください。

(XSLT 関数ではない)チャートに関する**XPath 関数**は、 2つのグループに分けられています：:

- チャートの生成と保存を行う関数
- チャートの作成を行う関数

メモ： チャート関数は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていないことに注意してください。

メモ： サーバーエディションのチャートでサポートされるイメージのフォーマットは`jpg`、`png`、および`bmp` です。無損失圧縮のため、推奨されるオプションは`png` です。Enterprise エディションでサポートされるフォーマットは`jpg`. `png`、`bmp`、および`gif` です。

チャートの生成と保存を行う関数

これらの関数は、 チャート作成関数により得られた) チャートオブジェクトを受け取り、イメージの生成を行うか、イメージをファイルへ保存します。

`altova:generate-chart-image` (`$chart, $width, $height, $encoding`) `as atomic`

以下の説明を参照ください

- `$chart` は`altova:create-chart` 関数により得られたチャート拡張アイテムです。
- `$width` ならびに`$height` により大きさを指定する必要があります。
- `$encoding` は`binarytobase64` または`binarytobase16` から選択することができます。

関数からは、指定されたエンコーディングならびにイメージフォーマットにてチャートのイメージが返されます。

`altova:generate-chart-image` (`$chart, $width, $height, $encoding, $imagetype`) `as atomic`

以下の説明を参照ください

- `$chart` は`altova:create-chart` 関数により得られたチャート拡張アイテムです。
- `$width` ならびに`$height` により大きさを指定する必要があります。
- `$encoding` は`base64Binary` または`hexBinary` であることができます。
- `$imagetype` は次のフォーマットから選択することができます：`png`、`gif`、`bmp`、`jpg`、`jpeg`。`gif` はサーバー製品でサポートされていないことに注意してください。*ページの上のメモを参照してください。*

関数からは、指定されたエンコーディングならびにイメージフォーマットにてチャートのイメージが返されます。

---

altova:save-chart-image ($chart, $filename, $width, $height) as empty()
(Windows のみ)

以下の説明を参照ください

- $chart はaltova:create-chart 関数により得られたチャート拡張アイテムです。
- $filename は保存されるチャートイメージのファイルパスならびにファイル名です。
- $width ならびに$height により大きさを指定する必要があります。

関数により$filename で指定されたファイルにチャートイメージが保存されます。

---

altova:save-chart-image ($chart, $filename, $width, $height, $imagetype) as empty() ***(Windows のみ)***

以下の説明を参照ください

- $chart はaltova:create-chart 関数により得られたチャート拡張アイテムです。
- $filename は保存されるチャートイメージのファイルパスならびにファイル名です。
- $width ならびに$height により大きさを指定する必要があります。
- $imagetype は次のフォーマットから選択することができます :png、gif、bmp、jpg、jpeg。gif はサーバー製品でサポートされていないことに注意してください。*ページの上のメモを参照してください。*

指定されたイメージフォーマットで$filename により指定されたファイルにチャートイメージが保存されます。

チャートの作成を行う関数

以下の関数によりチャートの作成を行うことができます

altova:create-chart($chart-config, $chart-data-series*) as chart extension item

以下の説明を参照ください

- $chart-config はaltova:create-chart-config 関数またはaltova:create-chart-config-from-xml 関数により取得されたchart-config 拡張アイテムとなります。
- $chart-data-series はaltova:create-chart-data-series 関数またはaltova:create-chart-data-series-from-rows 関数により取得されたchart-data-series 拡張アイテムとなります。

関数により、引数により与えられたデータから作成されたチャート拡張アイテムが返されます。

---

altova:create-chart-config($type-name, $title) as chart-config extension item

以下の説明を参照ください

- $type-name により、以下から作成される種類のチャートが選択されます：Pie、Pie3d、 BarChart、BarChart3d、BarChart3dGrouped、 LineChart、ValueLineChart、RoundGauge、BarGauge
- $title によりチャートの名前が指定されます。

関数により、チャートの構成情報が含まれるchart-config 拡張アイテムが返されます。

---

altova:create-chart-config-from-xml($xml-struct) as chart-config extension item

以下の説明を参照ください

- $xml-struct はチャートの構成情報が含まれるXML 構造です。

関数により、チャートの構成情報が含まれるchart-config 拡張アイテムが返されます。この情報はXML データフラグメントから与えられます。

---

altova:create-chart-data-series($series-name?, $x-values*, $y-values*) as chart-data-series extension item

以下の説明を参照ください

- $series-name により、系列の名前が指定されます。
- $x-values により、X-軸値のリストが与えられます。
- $y-values により、Y-軸値のリストが与えられます。

関数により、チャートの作成に必要なデータの情報（系列の名前と軸のデータ）が含まれるchart-data-series 拡張アイテムが返されます。

---

altova:create-chart-data-row(x, y1, y2, y3, ...) as chart-data-x-*N*y-row extension item

以下の説明を参照ください

- x はチャートデータ行のX-軸カラムにおける値です。
- yN はY-軸カラムの値です。

関数により、1つの系列のX-軸カラムならびにY-軸カラムに対するデータが含まれるchart-data-x-*N*y-row 拡張アイテムが返されます。

---

altova:create-chart-data-series-from-rows($series-names as xs:string*, $row*) as chart-data-series extension item

以下の説明を参照ください

- `$series-name` は作成される系列の名前です。
- `$row` は系列として作成される`chart-data-x-Ny-row` 拡張アイテムです。

関数により、系列のX-軸ならびにY-軸に関するデータが含まれる`chart-data-series` 拡張アイテムが返されます。

---

altova:create-chart-layer(`$chart-config, $chart-data-series*) as chart-layer extension item`

以下の説明を参照ください

- `$chart-config` は`altova:create-chart-config` 関数または`altova:create-chart-config-from-xml` 関数により取得された`$chart-config` 拡張アイテムです。
- `$chart-data-series` は`altova:create-chart-data-series` 関数または`altova:create-chart-data-series-from-rows` 関数により取得された`chart-data-series` 拡張アイテムです。

関数により、チャートのレイヤーデータが含まれる**`chart-layer`** 拡張アイテムが返されます。

---

altova:create-multi-layer-chart(`$chart-config, $chart-data-series*, $chart-layer*)`

以下の説明を参照ください

- `$chart-config` は`altova:create-chart-config` 関数または`altova:create-chart-config-from-xml` 関数により取得された`chart-config` 拡張アイテムです。
- `$chart-data-series` は`altova:create-chart-data-series` 関数または`altova:create-chart-data-series-from-rows` 関数により取得された`chart-data-series` 拡張アイテムです。
- `$chart-layer` は`altova:create-chart-layer` 関数により取得された**`chart-layer`** 拡張アイテムです。

関数により、複数レイヤーのチャートアイテムが返されます。

---

altova:create-multi-layer-chart(`$chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)`

以下の説明を参照ください

- `$chart-config` は`altova:create-chart-config` 関数または`altova:create-chart-config-from-xml` 関数により取得された`chart-config` 拡張アイテムです。
- `$chart-data-series` は`altova:create-chart-data-series` 関数または`altova:create-chart-data-series-from-rows` 関数により取得された`chart-data-`

series 拡張アイテムです。
- $chart-layer は altova:create-chart-layer 関数により取得された **chart-layer** 拡張アイテムです。

関数により、複数レイヤーのチャートアイテムが返されます。

チャートデータの XML 構造

以下にチャートデータの XML 構造、ならびにそれが Altova のチャート拡張関数においてどのように表示されるかを示します。チャートの種類によっては、ここでの記述内容により外観が変化します。全ての要素が全種類のチャートに対して使用されるわけではなく、例えば <Pie> 要素は棒グラフに対して無視されます。

メモ: チャート関数は、**Enterprise** ならびに**サーバー エディション**の Altova 製品でしかサポートされていないことにご注意してください。

```
<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="10pt" 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>
```

チャート関数

以下のサンプルXSLT ドキュメントでは、Altova のチャート拡張関数の使用方法が示されます。更に下には、XML ドキュメントと、Altova XSLT 2.0 エンジンとXSLT ドキュメントによりXML ドキュメントが処理された結果生成される出力イメージのスクリーンショットが示されます。

メモ： チャート関数は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていないことにご注意してください。

メモ： チャートデータテーブルの作成に関する詳しい情報については、Altova XMLSpy ならびにStyleVision 製品のドキュメンテーションを参照ください。

XSLT ドキュメント

以下にあるXSLT ドキュメントでは、Altova チャート拡張関数を使用することで円グラフを生成します。更に下に記されているXML ドキュメントを処理するために使用することができます。

```
<?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="32%"
                                    TitleToPlotMargin="3%"
                                    LegendToPlotMargin="6%"
                                    >
                                    <TitleFont
                                        Color="#023d7d"
                                        Name="Tahoma"
                                        Bold="1"
                                        Italic="0"
                                        Underline="0"
                                        MinFontHeight="10pt"
                                        Size="8%" />
                                </General>
                            </chart-config>
                        </xsl:variable>
                        <xsl:sequence select="altova:ext: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="altova:ext:create-chart-data-row( (@id), ( . ) )"/>
                            </xsl:for-each>
                        </xsl:variable>
                        <xsl:variable name="chartDataSeriesNames" as="xs:string*" select=" ( (&quot;Series
1&quot;), &apos;&apos; )[1]"/>
                        <xsl:sequence
                            select="altova:ext:create-chart-data-series-from-rows( $chartDataSeriesNames,
$chartDataRows)"/>
                    </xsl:variable>
                    <xsl:variable name="ChartObj" select="altova:ext:create-chart( $extChartConfig,
( $chartDataSeries), false() )"/>
                    <xsl:variable name="sChartFileName" select="'mychart1.png'"/>
                    <img src="{$sChartFileName, altova:ext:save-chart-image( $ChartObj, $sChartFileName,
400, 400 )}"/>
                </xsl:for-each>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

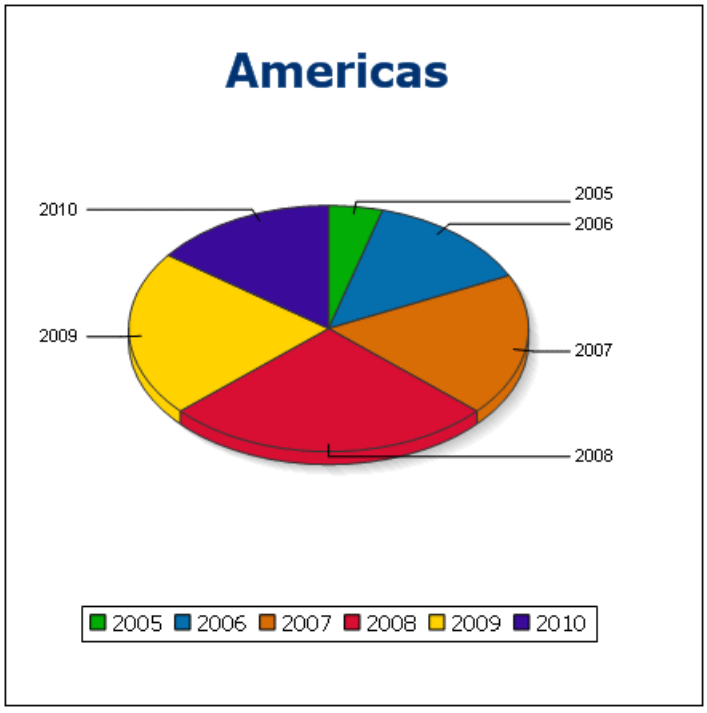
XML ドキュメント

上にあるXSLT ドキュメントにより以下のXML ドキュメントを処理することができます。XML ドキュメント内にあるデータを使用することで、下に示される円グラフを生成することができます。

```
<?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>
```

出力イメージ
上にあるXSLT ドキュメントを使ってXML ドキュメントを処理することで、以下にある円グラフが生成されます



19.1.2.2 その他の拡張関数

Java やC# などのプログラミング言語には、XPath 2.0 / XQuery 関数、またはXSLT 2.0 関数として利用できない関数がいくつかあります。そのような関数の良い例として、Java で利用することのできる`sin()`や`cos()`という数学関数があります。XSLT スタイルシートやXQuery のクエリにてこれらの関数が利用できるのであれば、スタイルシートやクエリの適用範囲を大幅に拡張することができ、スタイルシート作成タスクの負担が大幅に軽減されます。Altova 製品で使用されているAltova エンジン (XSLT 1.0、XSLT 2.0、XQuery 1.0) では、Java や.NET および MSXSL scripts for XSLTにおける拡張関数の使用がサポートされます。 このセクションでは、拡張機能およびXSLT スタイルシート内で

MSXSL スクリプト、および XQuery ドキュメントを使用する方法について記述します。使用できる拡張関数は以下のように構成されます：

- Java 拡張関数
- .NET 拡張関数
- XSLT に対する MSXSL スクリプト

記述の中では特に、(i) 関連するライブラリ内の関数がどのように呼ばれるか、(ii) 関数呼び出しを行う際に入力として使用される引数を変換するのにどのようなルールが適用され、return により値が返される際にどのような変換ルールが適用されるのか (XSLT/XQuery データオブジェクトに対する関数の結果) について説明されます。

必要条件

拡張関数のサポートを有効にするには、XSLT 変換や XQuery の実行を行うコンピューターに Java Runtime Environment (Java 関数にアクセスする場合) ならびに .NET Framework 2.0 以上 (.NET 関数にアクセスする場合) がインストールされている、またはアクセスできる環境が整っている必要があります。

Java 拡張関数

Java 拡張関数は、XPath または XQuery 条件式にて使用することができるほか、Java のコンストラクターを呼び出したり、Java の (静的または*インスタンス*) メソッドを呼び出すことができます。

Java *クラス*のフィールドは、引数を持たないメソッドとして扱われます。フィールドは静的または*インスタンス*として存在することができます。フィールドへの*アクセス*方法については、静的と*インスタンス*の両方について、以下のサブセクションにて記述されます。

このセクションは以下のサブセクションにより構成されます：

- Java：コンストラクター
- Java：静的メソッドと静的フィールド
- Java：インスタンスメソッドとインスタンスフィールド
- データ型：XPath/XQuery から Java へ
- データ型：Java から XPath/XQuery へ

拡張関数のフォーム

XPath/XQuery 条件式における拡張関数では、`prefix:fname()` の形式を取る必要があります。

- `prefix:` 部により拡張関数が Java 関数として認識されます。`java:` から始まる URI のスコープ内の名前空間宣言に拡張関数を関連付けることで Java 関数であるという認識が行われます。名前空間の宣言により、例えば `xmlns:myns="java:java.lang.Math"` という Java *クラス*が特定されます。名前空間の宣言は、(エラー無しの) `xmlns:myns="java"` という形式で、Java *クラス*の識別子を拡張関数にある `fname()` 部の左型に配置することでも行うことができます。
- `fname()` 部により、呼び出されている Java メソッドが識別され、メソッドの引数が提供されます (以下の例を参照ください)。`prefix:` 部にて識別された名前空間 URI が Java *クラス*を識別できなかった場合、Java *クラス*の識別は*クラス*の前にくる `fname()` 部にて行うことになり、ピリオドにより*クラス*から分離されることになります (以下にある2番目の XSLT サンプルを参照)。

メモ: 呼び出されるクラスはコンピューターのクラスパス上にある必要があります。

XSLT サンプル

以下に静的メソッドを呼び出す2つのサンプルを示します。最初のサンプルでは、クラス名 (`java.lang.Math`) が名前空間 URI に加えられており、`fname()` へ加えることはできません。2番目のサンプルでは、`prefix:` 部に`java:` が与えられており、`fname()` 部にてクラスとメソッドが識別されます。

```
<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)" />
```

拡張関数内にあるメソッド名 (上の例では`cos()`) は、名前付きJava クラス (上の例では`java.lang.Math`) のpublic な静的メソッドの名前に一致する必要があります。

XQuery サンプル

以下にXSLT のサンプルに似たXQuery のサンプルを示します :

```
<cosine xmlns:jMath="java:java.lang.Math">
  {jMath:cos(3.14)}
</cosine>
```

ユーザー定義された Java クラス

独自のJava クラスやメソッドを作成した場合、(i) JAR ファイル (またはclass ファイル) を介してこれらクラスファイルへアクセスしているか、(ii) これら (JAR またはclass) ファイルが、カレントディレクトリ (XSLT やXQuery ドキュメントが存在するディレクトリ) に配置されているかにより、これらクラスの呼び出し方法が変わってきます。これらファイルの特定方法については、ユーザー定義クラスファイルならびにユーザー定義 JAR ファイルを参照ください。カレントディレクトリには無いクラスファイルやJAR ファイルへのパスは指定しなければならないことにご注意してください。

ユーザー定義のクラスファイル

アクセスがクラスファイルを介したものである場合、4つのケースが考えられます :

- クラスファイルはパッケージである。XSLT またはXQuery ファイルがJava パッケージと同じ場所に収められている。 (下のサンプルを参照)
- クラスファイルはパッケージでは無い。XSLT またはXQuery ファイルがJava パッケージと同じ場所に収められている。 (下のサンプルを参照)
- クラスファイルはパッケージである。XSLT またはXQuery ファイルがランダムな場所に収められている。 (下のサンプルを参照)
- クラスファイルはパッケージである。XSLT またはXQuery ファイルがランダムな場所に収められている。 (下のサンプルを参照)

クラスファイルがパッケージではなく、XSLT または XQuery ドキュメントと同じ場所に収められているケースを考えてみましょう。この場合、フォルダー内の全クラスを発見することができるため、ファイルの場所を指定する必要はありません。クラスの識別を行う構文は以下のようになります：

```
java:classname
```

ここで、

`java:` によりユーザー定義の Java 関数が呼ばれていることが示されます（デフォルトでカレントディレクトリにある Java クラスがロードされます）。

`classname` は目的となるメソッドのクラスが含まれているクラスの名前です。

クラスが名前空間 URI にて識別され、名前空間がメソッド呼び出しにて使用されます。

クラスファイルがパッケージで、XSLT または XQuery ファイルが Java パッケージと同じ場所に収められている

以下の例では、`com.altova.extfunc` パッケージにある `Car` クラスの `getVehicleType()` メソッドが呼び出されています。com.altova.extfunc パッケージは `JavaProject` という名前のフォルダーに置かれており、XSLT ファイルも同じフォルダーに配置されています。

```
<xsl:stylesheet version="2.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:com.altova.extfunc.Car">
  <xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

  <xsl:template match="/">
    <a>
    <xsl:value-of select="car:getVehicleType()"/>
    </a>
  </xsl:template>

  </xsl:stylesheet>
```

クラスファイルがパッケージではなく、XSLT または XQuery ファイルが Java パッケージと同じ場所に収められている

以下の例では、`com.altova.extfunc` パッケージにある Car クラスの `getVehicleType()` メソッドが呼び出されています。Car クラスファイルは `JavaProject/com/altova/extfunc` 以下に配置されています。XSLT ファイルも `JavaProject/com/altova/extfunc` フォルダーに配置されています。

```
<xsl:stylesheet version="2.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:Car">
  <xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

  <xsl:template match="/">
    <a>
    <xsl:value-of select="car:getVehicleType()"/>
    </a>
```

```
</xsl:template>

</xsl:stylesheet>
```

クラスファイルがパッケージで、XSLT または XQuery ファイルがランダムな場所に収められている

以下の例では、com.altova.extfunc パッケージにある Car クラスの getVehicleColor() メソッドが呼び出されています。com.altova.extfunc パッケージは JavaProject という名前のフォルダーに置かれており、XSLT ファイルが任意の場所に配置されています。この場合、以下のような構文でパッケージの場所をクエリ文字列として URI 内にて指定する必要があります：

```
java:classname[?path=uri-of-package]
```

ここで、

java: によりユーザー定義の Java 関数が呼ばれていることを表します。
uri-of-package は Java パッケージの URI です。
classname は目的のメソッドが含まれているクラス名です。

クラスは名前空間 URI により特定され、名前空間がメソッド呼び出しのプレフィックスで使用されます。以下の例ではカレントディレクトリ以外にあるクラスファイルへのアクセスを行うことができます。

```
<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 xs"/>

<xsl:template match="/">
  <xsl:variable name="myCar" select="car:new('red')" />
  <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

クラスファイルがパッケージではなく、XSLT または XQuery ファイルがランダムな場所に収められている

以下の例では、com.altova.extfunc パッケージにある Car クラスの getCarColor() メソッドが呼び出されています。com.altova.extfunc パッケージは JavaProject という名前のフォルダーに置かれており、XSLT ファイルが任意の場所に配置されています。以下のような構文で、クラスファイルの場所をクエリ文字列として URI 内にて指定する必要があります:

```
java:classname[?path=uri-of-classfile]
```

ここで

java: によりユーザー定義の Java 関数が呼ばれていることを表します。
uri-of-classfile は Java パッケージの URI です。
classname は目的のメソッドが含まれているクラス名です。

クラスは名前空間 URI により特定され、名前空間はメソッド呼び出しのプレフィックスで使用されます。以下の例で

はカレントディレクトリ以外にあるクラスファイルへのアクセスを行うことができます。

```
<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 xs xsl"/>

<xsl:template match="/">
  <xsl:variable name="myCar" select="car:new('red')" />
  <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

メモ: パスが外部関数により与えられている場合、ClassLoader によりパスが追加されます。

ユーザー定義の JAR ファイル

JAR ファイル経由でアクセスが行われた場合、以下の構文によりJAR ファイルのURI を指定する必要があります：

```
xmlns:classNS="java:classname?path=jar:uri-of-jarfile!/"
```

クラスの識別を行う名前空間 URI のプレフィックスを使用してメソッドが呼び出されます：`classNS:method()`

上の例に対する説明は以下のとおりです:

`java:` 関数が呼び出されていることを表します。
`classname` がユーザー定義されたクラスの名前になります。
`?` はクラス名とパスを分離するために使用されます。
`path=jar:` により、JAR ファイルへのパスが与えられていることを示します。
`uri-of-jarfile` はJAR ファイルのURI となります。
`!/` は、終了を表すデリミタとなります。
`classNS:method()` により、メソッドの呼び出しが行われます。

その他にも、メソッド名とともにクラス名を与えることができます。構文の例を以下に示します：

```
            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()
```

以下にJAR ファイルを使ったJava 拡張関数を呼び出すXSLT サンプルを記します：

```
<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 xs xsl"/>
```

```
<xsl:template match="/">
        <xsl:variable name="myCar" select="car:Car1.new('red')" />
        <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>

</xsl:stylesheet>
```

メモ： 拡張関数にクラスパスが与えられている場合、ClassLoader にパスが追加されます。

Java：コンストラクター

拡張関数を使用することで Java コンストラクターを呼び出すことができます。`new()` により全てのコンストラクターを呼び出すことができます。

Java コンストラクターの呼び出し結果を、默示的に XPath/XQuery データ型へ変換できる場合、Java 拡張関数により XPath/XQuery データ型のシーケンスが返されます。Java コンストラクターの呼び出し結果が XPath/XQuery データ型へ変換できない場合、値を返すクラス名でラップした Java オブジェクトがコンストラクターにより作成されます。例えば、`java.util.Date` クラスに対するコンストラクターが呼び出された場合（`java.util.Date.new()`）、`java.util.Date` を持ったオブジェクトが返されます。返されたオブジェクトのレキシカルフォーマットは、XPath データ型のレキシカルフォーマットにマッチしない場合もあり、目的の XPath データ型に対するレキシカルフォーマットへ値の変換を行い、その後目的の XPath データ型へ変換を行う必要があります。

コンストラクターにより作成された Java オブジェクトにより 2 つのことが行えます：

- 変数への割り当てを行うことができます：
  `<xsl:variable name="currentdate" select="date:new()" xmlns:date="java:java.util.Date" />`
- 拡張関数への受け渡しを行うことができます（*インスタンスメソッドならびにインスタンスフィールド*を参照ください）：
  `<xsl:value-of select="date:toString(date:new())" xmlns:date="java:java.util.Date" />`

Java：静的メソッドと静的フィールド

静的メソッドは、Java 名ならびにメソッドの引数により直接呼び出すことができます。`E` や `PI` といった定数の静的フィールド（引数を持たないメソッド）は、引数を指定することなくアクセスすることができます。

XSLT の例

静的メソッドならびにフィールドを呼び出す例を以下に示します：

```
<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)" />
```

上の拡張関数は `prefix:fname()` という形式を使用していることに注意してください。3 つの例にあるプレフィックスは全

てjMath: となっており、このプレフィックスは、java:java.lang.Math という名前空間 URI に関連付けられています。名前空間 URI は java: で開始しなければなりません。上の例では、クラス名 (java.lang.Math) を含むように拡張されています。拡張関数の fname() 部は (java.lang.Math のような) public クラスにマッチする必要があり、その後 public な静的メソッドが引数とともに続くか (例：cos(3.14))、public な静的フィールドが続きます (例：PI())。

上の例では、クラス名が名前空間 URI に含まれています。クラス名が名前空間 URI に含まれていない場合、以下の例にあるように、拡張関数の fname() 部にて追加する必要があります：

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math:cos(3.14)" />
```

XQuery の例

XQuery における似たようなサンプルを以下に示します：

```
<cosine xmlns:jMath="java:java.lang.Math">
  {jMath:cos(3.14)}
</cosine>
```

Java：インスタンスメソッドとインスタンスフィールド

メソッド呼び出しの第一引数としてパースされる Java オブジェクトが、インスタンスメソッドには与えられています。このような Java オブジェクトは通常、拡張関数を使うことで作成される (例：コンストラクターの呼び出し) か、スタイルシートパラメーター変数により作成されます。以下に XSLT サンプルを示します：

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
    <enrollment institution-id="Altova School"
          date="{date:toString($CurrentDate)}"
          type="{jlang:Object.toString(jlang:Object.getClass(date:new()))}">
    </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

上の例では、ノード enrollment/@type の値が以下のように作成されます：

1. java.util.Date クラスに対するオブジェクトがコンストラクター (date:new() コンストラクター) とともに作成されます。
2. jlang.Object.getClass メソッドの引数として Java オブジェクトがパースされます。
3. getClass メソッドにより得られたオブジェクトが jlang.Object.toString メソッドの引数としてパースされます。

結果 (@type の値) は java.util.Date を持った文字列となります。

インスタンスフィールドは、引数としてインスタンスフィールドへ渡される Java オブジェクトではないという点で、理論的にはインスタンスメソッドと異なります。パラメーターや変数がその代わりに引数として渡されますが、パラメーター変数そのものに Java オブジェクトから返された値が含まれている場合もあります。例えば、CurrentDate パラメーターには java.util.Date クラスのコンストラクターから返された値が含まれます。この値は引数として、date:toString インスタンスメソッドへ渡され、/enrollment/@date の値として使用されます。

データ型：XPath/ XQuery から Java へ

XPath/XQuery 条件式内部からJava 関数が呼び出された場合、複数ある同名のJava クラスのうち、どのクラスが呼び出されたのか決定するのに、関数へ渡される引数のデータ型が重要になります。

Java では、以下のルールが適用されます：

- 同名のJava メソッドが2つ以上あり、それぞれが違う数の引数を受け取る場合、呼び出しに使用されている引数の数に一番マッチするメソッドが選択されます。
- XPath/XQuery の文字列、数値、boolean データ型は、黙示的に対応するJava データ型へ変換されます（以下のリストを参照）。与えられたXPath/XQuery 型が2つ以上のJava 型へ変換できる場合（例：`xs:integer`）、選択されたメソッドにて宣言されているJava 型が使用されます。例えば、呼び出されたJava メソッドが `fx(decimal)` で、与えられたXPath/XQuery データ型が`xs:integer` の場合、`xs:integer` がJava の`decimal` データ型へ変換されます。

以下のテーブルに、XPath/XQuery の文字列、数値、boolean 型からJava データ型への黙示的な変換リストを示します。

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (プリミティブ型), `java.lang.Boolean`
`xs:integer`	`int, long, short, byte, float, double,` ならびに `java.lang.Integer` のようなこれらのラッパークラス
`xs:float`	`float` (プリミティブ型),`java.lang.Float`, `double` (プリミティブ型)
`xs:double`	`double` (プリミティブ型), `java.lang.Double`
`xs:decimal`	`float` (プリミティブ型), `java.lang.Float`, `double`(プリミティブ型), `java.lang.Double`

上のリストにあるXML スキーマデータ型（ならびにXPath やXQuery で使用されているデータ型）のサブタイプも、対応する祖先のサブタイプとしてJava のデータ型へ変換されます。

場合によっては、与えられた情報から正しいJava メソッドを選択することができない場合もあります。例えば、以下のような場合を考えてみましょう：

- 与えられた引数が 10 という値を持ったxs:untypedAtomic 型で、`mymethod(float)` メソッドへ渡されるのを意図している。
- しかし、そのクラスには別のデータ型を取る`mymethod(double)` というメソッドも存在する。
- メソッド名が同じで、与えられた型（`xs:untypedAtomic`）も`float` と`double` の両方に変換することができるため、`xs:untypedAtomic` が`float` ではなく`double` に変換される可能性もある。
- 結果として、意図したメソッドは選択されず、予期しない動作結果が招く可能性がある。この問題を回避するには、意図したメソッドを使用するユーザー定義のメソッドを別の名前で新たに作成する必要があります。

上のリストでカバーされていない型（例：`xs:date`）は変換されず、エラーとなります。しかし場合によっては、Java コンストラクターを使用して、目的のJava データ型を作成することが可能だということも留意してください。

データ型：Java から XPath/ XQuery へ

Java メソッドにより値が返され、値のデータ型が文字列、数値、または boolean 型の場合、対応する XPath/XQuery 型への変換が行われます。例えば、Java の `java.lang.Boolean` や `boolean` データ型は `xsd:boolean` へ変換されます。

関数から返された一次元配列は、シーケンス (sequence) に展開されます。2次元以上の配列は変換されることが無いため、ラップして使用するべきでしょう。

Java オブジェクトや文字列、数値、boolean 以外のデータ型がラップされて返された場合、最初に (例えば `toString` という) Java メソッドを使用して Java オブジェクトを文字列へ変換することで、目的の XPath/XQuery 型への変換を行います。XPath/XQuery では、文字列を目的となる型のレキシカルフォーマットへ変換し、目的の型への変換を (例えば `cast as` 式を使用することで) 行うことができます。

.NET 拡張関数

.NET プラットフォームにて作業を行なっている場合、.NET 言語 (例えば C#) で記述された拡張関数を使用することができます。.NET 拡張関数は XPath や XQuery 条件式内部から使用することができ、.NET クラス内部にあるコンストラクターや (static または *インスタンス* 変数) プロパティを呼び出すことができます。

`get_PropertyName()` 構文を使用することにより .NET クラスのプロパティを呼び出すことができます。

このセクションは、以下のサブセクションにより構成されています：

- .NET コンストラクター
- .NET 静的メソッドならびに静的フィールド
- .NET：インスタンスメソッドとインスタンスフィールド
- データ型：XPath/XQuery から .NET へ
- データ型：.NET から XPath/XQuery へ

拡張関数のフォーム

XPath/XQuery 条件式にある拡張関数は、`prefix:fname()` の形式を取る必要があります。

- `prefix:` 部は、呼び出されている .NET クラスを特定する URI となります。
- `fname()` 部により、.NET クラス内にあるコンストラクター、プロパティ、または (静的または *インスタンス*) メソッドが特定され、必要な場合は引数が与えられます。
- URI は `clitype:` で開始する必要があり、これにより関数が .NET 拡張関数であることが認識されます。
- 拡張関数の `prefix:fname()` 形式は、*システムクラス* ならびにロードされたアセンブリとともに使用することもできます。しかしクラスをロードする必要がある場合、必要な情報が含まれるパラメーターが必要になります。

パラメーター

アセンブリをロードするには以下のパラメーターを使用してください：

`asm` ロードするアセンブリの名前。

ver	バージョン番号（ピリオドにより分離された最大 4 桁の整数）。
sn	アセンブリ厳密名のキートークン（16 新数の数値）。
from	ロードするアセンブリ (DLL) の場所を特定する URI。URI が相対パスの場合、XSLT や XQuery ドキュメントに対して相対的になります。このパラメーターが指定された場合、その他のパラメーターが無視されます。
partialname	アセンブリ名の一部。`Assembly.LoadWith.PartialName()` へ渡され、アセンブリへのロードが試みられます。`partialname` が指定された場合、その他のパラメーターが無視されます。
loc	例えば `en-US` というロケール。デフォルトは `neutral` です。

アセンブリが DLL からロードされる場合、`from` パラメータが使用して、`sn` パラメーターは使用しないでください。アセンブリがグローバルアセンブリキャッシュ (GAC) からロードされる場合、`sn` パラメータを使用して `from` パラメーターは使用しないでください。

最初のパラメーターの前に疑問符 (?) を挿入し、パラメーター同士はセミコロンで分離する必要があります。パラメーター名へ値を受け渡すには、統合符号 (=) を使用します（以下の例を参照ください）。

名前空間宣言の例

XSLT において、*システムクラス* `System.Environment` を特定する名前空間宣言の例を以下に示します：

```
xmlns:myns="clitype:System.Environment"
```

XSLT において、ロードする*クラス*を `Trade.Forward.Scrip` として特定する名前空間宣言の例を以下に示します：

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

XQuery において、*システムクラス* `MyManagedDLL.testClass` を特定する名前空間宣言の例を以下に示します。2 つのケースが考えられます：

1. アセンブリが GAC からロードされた場合：

```
declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
        ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
```

2. アセンブリが DLL からロードされた場合（完全参照と一部の参照）：

```
        declare namespace cs="clitype:MyManagedDLL.testClass?
from=file:///C:/Altova
        Projects/extFunctions/MyManagedDLL.dll;

    declare namespace cs="clitype:MyManagedDLL.testClass?
    from=MyManagedDLL.dll;
```

XSLT の例

システムクラス `System.Math` 内の関数を呼び出すための完全な XSLT の例を以下に示します：

```
<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>
```

math 要素にある名前空間宣言により、math: プレフィックスとclitype:System.Math URI が関連付けられます。URI の最初にある clitype: により、それ以降の記述がシステムクラスまたはロードされたクラスを特定するものであることが示されます。XPath 条件式にあるmath: プレフィックスにより、拡張関数がURI (そしてクラス) System.Math に関連付けられます。拡張関数により、System.Math クラス内のメソッドが特定され、必要な場所に引数が与えられます。

XQuery の例

上のXSLT に対する例と同様のXQuery 例を以下に示します：

```
<math xmlns:math="clitype:System.Math">
  {math:Sqrt(9)}
</math>
```

上のXSLT と同様に、名前空間宣言により NET クラス (この場合はシステムクラス) が特定されます。XQuery 式により呼び出されるメソッドが特定され、引数が与えられます。

.NET コンストラクター

拡張関数を使用することで、NET コンストラクターを呼び出すことができます。new() により全てのコンストラクターを呼び出すことができます。クラス内に2つ以上のコンストラクターがある場合、与えられた引数の数が最もマッチするコンストラクターが選択されます。与えられた引数に対してマッチするコンストラクターが見つからない場合、'No constructor found'" エラーが返されます。

XPath/XQuery データ型を返すコンストラクター

.NET コンストラクター呼び出しの結果が XPath/XQuery データ型へ黙示的に変換することができる場合、NET 拡張関数からXPath/XQuery データ型のシーケンスが返されます。

.NET オブジェクトを返すコンストラクター

.NET コンストラクター呼び出しの結果がXPath/XQuery データ型へ適切に変換できない場合、値を返すクラス名でラップした NET オブジェクトがコンストラクターにより作成されます。例えば、System.DateTime クラスのコンストラクターが (System.DateTime.new() により) 呼ばれた場合 System.DateTime 型を持ったオブジェクトが返されます。

返されたオブジェクトのレキシカルフォーマットは、目的のXPath データ型と違っている場合があります。その場合、返された値を：(i) 目的のXPath データ型のレキシカルフォーマットへ変換し、(i) 目的のXPath データ型へキャストする必要があります。

コンストラクターにより作成された NET オブジェクトに対して3つのことを行うことができます：

- 変数内で使用することができます：
```
<xsl:variable name="currentdate" select="date:new(2008,4,29)"
xmlns:date="clitype:System.DateTime" />
```
- 拡張関数へ渡すことができます（*インスタンスメソッドとインスタンスフィールド*を参照ください）：
```
<xsl:value-of select="date:ToString(date:new(2008,4,29))"
xmlns:date="clitype:System.DateTime" />
```
- 文字列、数値、または boolean へ変換することができます：
```
<xsl:value-of select="xs:integer(date:get_Month(date:new(2008,4,29)))"
xmlns:date="clitype:System.DateTime" />
```

.NET：静的メソッドと静的フィールド

メソッド名と引数を与えることで、静的メソッドを直接呼び出すことができます。呼び出しに使用される名前は、クラス内にある public static メソッドと完全に一致する必要があります。関数の呼び出しに使用されたメソッド名と引数の数にマッチするものがクラス内に複数ある場合、与えられた引数が評価され、最もマッチするものが選択されます。マッチする結果が得られない場合、エラーが返されます。

メモ： .NET クラス内にあるフィールドは、引数を持たないメソッドとしてみなされます。プロパティは `get_PropertyName()` 構文により呼び出されます。

例

1つの引数とともにメソッド（`System.Math.Sin(arg)`）を呼び出す XSLT サンプルを以下に示します：

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

引数なしのメソッドとしてみなされる）フィールド（`System.Double.MaxValue()`）を呼び出す XSLT サンプルを以下に示します：

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

（`get_PropertyName()` 構文を使って）プロパティ（`System.String()`）を呼び出す XSLT サンプルを以下に示します：

```
<xsl:value-of select="string:get_Length('my string')" xmlns:string="clitype:System.String"/>
```

1つの引数とともにメソッド（`System.Math.Sin(arg)`）を呼び出す XQuery サンプルを以下に示します：

```
<sin xmlns:math="clitype:System.Math">
  {math:Sin(30)}
</sin>
```

.NET：インスタンスメソッドとインスタンスフィールド

インスタンスメソッドには、メソッド呼び出しの第一引数として .NET オブジェクトが渡されます。通常この .NET オブジェクトは、拡張関数（例えばコンストラクター呼び出し）またはスタイルシートパラメーター変数により作成されます。以下に XSLT の例を示します：

```
<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>
```

上の例では、System.DateTime コンストラクター (`new(2008, 4, 29)`) が、System.DateTime 型の .NET オブジェクトの作成に使用されます。このオブジェクトは、最初に `releasedate` 変数の値として、次に `System.DateTime.ToString()` メソッドの引数として作成されます。`System.DateTime.ToString()` インスタンスメソッドは、`System.DateTime` コンストラクターの (new(2008, 4, 29)) における引数として2度呼び出されます。これらインスタンスにおいて `releasedate` 変数が .NET オブジェクトを取得するのに使用されます。

インスタンスメソッドとインスタンスフィールド

インスタンスメソッドとインスタンスフィールドの違いは理論的なものです。インスタンスメソッドでは .NET オブジェクトが直接引数に渡され、インスタンスフィールドではパラメーターや変数が (.NET オブジェクトそのものを含めることはできるものの) 代わりに渡されます。例えば上の例では、`releasedate` 変数に .NET オブジェクトが含まれており、この変数が2番目の date 要素コンストラクターにて `ToString()` の引数として渡されます。そのため、最初の date 要素にある `ToString()` インスタンスがインスタンスメソッドであるのに対し、2番目はインスタンスフィールドとしてみなされます。両方のインスタンスで求められる結果は等価です。

データ型 : XPath/ XQuery から .NET へ

.NET 拡張関数が XPath/XQuery 条件式内部で使用された場合、複数ある .NET メソッドのうち、どれが呼び出されたのか決定するのに関数の引数に使用されるデータ型が重要になります。

.NET では、以下のルールが適用されます :

- クラス内に同名のメソッドが2つ以上ある場合、呼び出しに使用された引数の数がマッチするメソッドだけが、呼び出される関数の候補に絞められます。
- XPath/XQuery の文字列、数値、boolean データ型は黙示的に対応する .NET データ型へ変換されます (以下のリストを参照)。与えられた XPath/XQuery 型が2つ以上の .NET 型へ変換できる場合 (例 : `xs:integer`)、選択されたメソッドにて宣言されている .NET 型が使用されます。例えば、呼び出された .NET メソッドが fx(fouble) で、与えられた XPath/XQuery データ型が xs:integer の場合、`xs:integer` が .NET の `double` データ型へ変換されます。

以下のテーブルに、XPath/XQuery の文字列、数値、boolean 型から .NET データ型へ行われる黙示的な変換リストを示します。

xs:string	StringValue, string
xs:boolean	BooleanValue, bool
xs:integer	IntegerValue, decimal, long, integer, short, byte, double, float
xs:float	FloatValue, float, double
xs:double	DoubleValue, double
xs:decimal	DecimalValue, decimal, double, float

上のリストにあるXML スキーマ型 ならびにXPath やXQuery で使用されているデータ型) のサブタイプも、対応する祖先のサブタイプとして .NET のデータ型へ変換されます。

場合によっては、与えられた情報から正しい.NET メソッドを選択することができない場合もあります。例えば、以下のような場合を考えてみましょう:

- 与えられた引数が10ある`xs:untypedAtomic` 値で、`mymethod(float)` メソッドへ渡されるのを意図している。
- しかし、そのクラスには別のデータ型をとる`mymethod(double)` というメソッドも存在する。
- メソッド名が同じで、与えられた型 (`xs:untypedAtomic`) も`float` と`double` の両方に変換することができるため、`xs:untypedAtomic` が`float` ではなく`double` に変換される可能性もある。
- 結果として、意図したメソッドは選択されず、予期しない動作結果が招く可能性がある。この問題を回避するには、意図したメソッドを使用するユーザー定義のメソッドを別の名前で新たに作成する必要があります。

上のリストでカバーされていない型 (例:`xs:date`) は変換されずエラーとなります。

データ型:.NET から XPath/ XQuery へ

.NET メソッドにより値が返される際に値のデータ型が文字列、数値、またはboolean 型の場合、対応するXPath/ XQuery 型への変換が行われます。例えば、 .NET の`decimal`データ型は`xsd:decimal` へ変換されます。

.NET オブジェクトや文字列、数値、boolean 以外のデータ型が返された場合、最初に (例えば `System.DateTime.ToString()` という) .NET メソッドを使用して .NET オブジェクトを文字列へ変換します。XPath/XQuery では、文字列を目的となる型のレキシカルフォーマットへ変換し、目的の型への変換を (例えば`cast as` 式を使用することで) 行うことができます。

XSLT に対する MSXSL スクリプト

`<msxsl:script>` 要素にはユーザー定義の関数や変数が含まれており、XSLT スタイルシート内のXPath 条件式内部から呼び出しを行うことができます。`<msxsl:script>` はトップレベル要素で、`<xsl:stylesheet>` または`<xsl:transform>` の子要素である必要があります。

`<msxsl:script>` 要素は`urn:schemas-microsoft-com:xslt` 名前空間内に存在する必要があります (以下を参照ください)。

スクリプト言語と名前空間

ブロック内で使用されるスクリプト言語は<msxsl:script> 要素の language 属性にて指定され、XPath 条件式における関数の呼び出しに対して使用される名前空間は implements-prefix 属性により特定されます (以下を参照)。

```
<msxsl:script language="scripting-language" implements-prefix="user-namespace-
prefix">

   function-1 or variable-1
   ...
   function-n or variable-n

</msxsl:script>
```

<msxsl:script> 要素はWindows Scripting Runtime を使ったやりとりを行うため、お使いのコンピューターにインストールされた言語だけしか<msxsl:script> 要素では使用することができません。**MSXSL スクリプトを使用するには .NET Framework 2.0 以上のプラットフォームをインストールする必要があります。結果として** <msxsl:script> 言語から .NET スクリプト言語を使用することができます。

HTML の<script> 要素における language 属性と同じ値が language 属性では受理されます。language 属性が指定されていない場合、Microsoft JScript がデフォルトとして想定されます。

implements-prefix 属性には名前空間スコープ内で宣言されたプレフィックスが与えられます。通常この名前空間は関数ライブラリのために予約されたユーザーの名前空間となります。<msxsl:script>要素内で定義された全ての関数ならびに変数は、implements-prefix 属性にて指定されたプレフィックスで特定される名前空間に収められます。XPath 条件式内部から関数が呼ばれる場合、完全修飾関数名が同じ名前空間内に関数として定義されていなければなりません。

例

<msxsl:script> 要素内で定義された関数を使用するXSLT スタイルシートの例を以下に示します：

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:fn="http://www.w3.org/2005/xpath-functions"
  xmlns:msxsl="urn:schemas-microsoft-com:xslt"
  xmlns:user="http://mycompany.com/mynamespace">

 <msxsl:script language="VBScript" implements-prefix="user">
  <![CDATA[
  'Input: A currency value: the wholesale price
  'Returns: The retail price: the input value plus 20% margin,
  'rounded to the nearest cent
  dim a as integer = 13
  Function AddMargin(WholesalePrice) as integer
    AddMargin = WholesalePrice * 1.2 + a
  End Function
 ]]>
 </msxsl:script>

 <xsl:template match="/">
  <html>
   <body>
    <p>
     <b>TotalRetailPrice =
      $<xsl:value-of select="user:AddMargin(50)"/>
     </b>
```

```
        <br/>
        <b>TotalWholesale Price =
          $<xsl:value-of select="50"/>
        </b>
      </p>
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

データ型

スクリプトブロックとのやりとりに使用されるパラメーターの値はXPath データ型に限定されます。スクリプトブロック内にある関数にでやりとりされるデータや変数に、この制限はありません。

アセンブリ

msxsl:assembly 要素を使用することで、アセンブリをスクリプト内部へインポートすることができます。アセンブリは名前やURI により特定されます。アセンブリのインポートは、コンパイル時に行われます。以下に **msxsl:assembly** 要素の簡単な使用例を示します：

```
<msxsl:script>
       <msxsl:assembly name="myAssembly.assemblyName" />
       <msxsl:assembly href="pathToAssembly" />

       ...

</msxsl:script>
```

アセンブリ名は、以下のような完全な名前でも：:

```
"system.Math, Version=3.1.4500.1 Culture=neutral
PublicKeyToken=a46b3f648229c514"
```

"myAssembly.Draw" のような短い名前でも指定できます。

名前空間

msxsl:using 要素により名前空間の宣言を行うことができます。これにより、スクリプト内において名前空間無しでアセンブリクラスを使用することができ、タイピングの手間を軽減することができます。以下に msxsl:using 要素の簡単な使用例を示します：

```
<msxsl:script>
       <msxsl:using namespace="myAssemblyNS.NamespaceName" />

       ...

</msxsl:script>
```

namespace 属性の値は名前空間の名前となります。

19.2 技術データ

このセクションは、ソフトウェアの技術面に関する役に立つ背景情報を含んでいます。以下のように整理されています：

- OS とメモリ要件
- Altova XML バリデーター
- Altova XSLT とXQuery エンジン
- Unicode のサポート
- インターネットの使用

19.2.1 OS とメモリ要件

オペレーティングシステム
Altova ソフトウェア アプリケーションは、以下のプラットフォームでご使用いただけます：

- Windows XP/Vista, Windows 7/8/10
- Windows Server 2003/2008/2012/2016

メモリ
ソフトウェアがC++ で書かれているため、Java Runtime Environment をダウンロードする必要はなく、Java ベースのアプリケーションに比べ、通常少ないメモリを必要とします。しかしながら、各ドキュメントは完全に解析するため、また、ビューと編集の速度を向上するためにメモリにダウンロードされます。メモリの要件は、ドキュメントのサイズを増やします。

メモリ要件は、制限のない「元に戻す」履歴により影響を受けます。大きなドキュメントの大きなセクションの切り取り、貼り付け操作を繰り返し行うと、使用できるメモリがすぐに消費されます。

19.2.2 Altova XML バリデーター

XML ドキュメントを開くと、アプリケーションは、内蔵のXML バリデーターを使用して、指定されている場合、スキーマに対して整形式をチェックし、ツリーとインフォセットを作成します。XML バリデーターは、ドキュメントを編集する際にインテリジェントな編集ヘルプを提供し、発生する検証エラーを表示するために使用されます。

内蔵のXML バリデーターは、W3C のXML スキーマ 1.0 と 1.1 仕様の最終勧告を実装しています。 .New developments recommended by the W3C XML スキーマ作業グループにより勧告される新しい項目は、XML バリデーターに継続的に組み込まれるため、Altova 製品は最高水準の開発環境を届けることができます。

19.2.3 Altova XSLT と XQuery エンジン

Altova 製品 は、Altova XSLT 1.0、 2.0、 および 3.0 エンジンとAltova XQuery 1.0 と 3.1 エンジンを使用しています。各エンジンのためのドキュメントと実装に固有の振る舞いに関しては、製品で使用されるエンジンの各ドキュメントの付属書 (エンジン情報) で確認することができます。

メモ: Altova MapForce は、XSLT 1.0、2.0 およびXQuery 1.0 エンジンを使用したコードを生成します。

19.2.4 Unicode のサポート

Altova XML 製品は、Unicode を完全にサポートします。XML ドキュメントを編集するには、ドキュメント内で使用されているUnicode 文字をサポートするフォントが必要です。

フォントの多くは、Unicode 範囲全体の特定のサブセットを含む場合があり、このため、通常は対応する表記システムをターゲットとします。テキストの一部が、文字化けして表示された場合、理由としては、選択されたフォントが必要とする字形を含まない場合があげられます。ですから、特に、異なる言語、または、異なる言語システムのXML ドキュメントを編集する場合、範囲全体をカバーするフォントを使用することが役にたちます。典型的なUnicode フォントは、Windows PC の Arial Unicode MS で確認することができます。

アプリケーションフォルダーの /Examples フォルダー内で、異なる言語システムで表記された次の文章を含むUnicodeUTF-8.htmlというXHTML ファイルを確認してください:

- *When the world wants to talk, it speaks Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。)

XHTML ファイルを開くと、Unicode の可能性を確認することができ、使用中のPC の使用することのできるフォントによりサポートされている表記システムが表示されます。

19.2.5 インターネットの使用

Altova アプリケーションは、次の状況でインターネット接続を開始します:

- 登録ダイアログ (**「ヘルプ| ソフトウェアのライセンス認証」**)内の「評価キーコードをリクエスト」をクリックした場合、登録ダイアログボックス内の3つのフィールドが通常のhttp (ポート 80)接続を使用し、サーバーに転送され、無料の評価キーが顧客に通常のSMTP 電子メールを使用して送り返されます。
- Altova 製品の一部では、インターネットからファイルを開くことができます (**「ファイル| 開く| URL に切り替える」**)。この場合、ドキュメントは、次のプロトコルメソッドと接続の1つを使用して取得されます:HTTP (通常、ポート 80)、FTP (通常、ポート 20/21)、HTTPS (通常、ポート 443)。HTTP サーバーをポート 8080 で作動することもできます (URL ダイアログ内で、サーバー名とコロンの後にポートを指定します)。
- XML スキーマ、または、DTD を参照するXML ドキュメント、と、URL により指定されているドキュメントを開くと、参照されているスキーマドキュメントは、HTTP 接続 (ポート 80)またはURL により指定されている他のプロトコール (上のポイント 2参照)により抽出されます。XML ファイルが検証されている場合、スキーマドキュメントも抽出されます (オプションダイアログのファイルタブ内の (**「ツール| オプション」**))。アプリケーションに命令している場合、ドキュメントが開かれると検証が自動的に行われる場合もあります。
- WSDL とSOAP を使用するAltova アプリケーションでは、Web サービスを使用する接続は、WSDL ドキュメントにより定義されています。
- XMLSpy 内で、**「電子メールで送信」** コマンドを使用する場合、(**「ファイル| 電子メールで送信」**) 現在選択されている範囲、または、ファイルは、ユーザーのマシンにインストールされているMAPI コンプライアント電子メールプログラムにより送信されます。
- ソフトウェアの荒いセンス認証 とLiveUpdate の一部として、Altova ソフトウェア使用許諾書内で更に詳しい説明を確認することができます。

19.3 ライセンス情報

ライセンス情報

- ソフトウェアの配布 に関する情報
- ソフトウェアのアクティベーションとライセンスの計測 に関する情報
- このソフトウェアに関連する知的所有権 に関する情報
- このソフトウェア製品の使用に関するエンドユーザー使用許諾契約書

本製品を使用する前に、上記の情報をよくお読みください。ソフトウェアのインストール時に上記のすべての条件に同意したとみなされ、お客様は上記の条件に拘束されることを同意したとみなされます。

19.3.1 電子的なソフトウェアの配布

この製品は電子的なソフトウェアの配布により利用することが可能で、この配布方法により、以下のユニークなメリットがあります：

- 購入を決定する前に、無料でソフトウェアを試用することができます。
- ソフトウェアの購入を決定した際には、Altova Web サイトにて注文を行います。すぐにライセンス登録された製品の使用を開始することができます。
- オンラインにて注文を行うと、常に最新のソフトウェアをご利用いただけます。
- 製品パッケージには包括的なヘルプシステムが画面上に表示されます。最新バージョンのユーザーマニュアルは www.altova.com 上にあり、(i) HTML フォーマットによる閲覧、ならびに (ii) PDF フォーマットのダウンロードと印刷に対応しております。

30日間の評価期間
この製品をダウンロードした後は、最大で30日の間無料で製品の評価を行うことができます。20日間を超えた頃から、製品がライセンス登録されていないことがソフトウェアにより表示されます。このメッセージはアプリケーションが起動されるたびに表示され、30日間を超えてプログラムを使用するには、キーコードとして送られるAltova ソフトウェア使用許諾契約書 を購入し、ソフトウェアアクティベーションダイアログにて入力し、製品をアンロックする必要があります。ライセンスの購入はAltova Web サイトのオンラインショップにてお求めいただけます。

組織内でソフトウェアの評価を行う
評価版のソフトウェアを組織内のネットワークにて配布したい場合、またはインターネットに接続されていないコンピューターにてソフトウェアを使用する場合、どのような状態でも改変されていないことを条件に、セットアッププログラムだけの配布を行うことが可能です。ソフトウェアインストーラーへアクセスした人は、例外なく30日間の評価ライセンスキーコードをリクエストして、試用期間が経過した後は、製品を使い続けるためにライセンスの購入を行う必要があります。

詳細については、このセクション最後にあるAltova ソフトウェアライセンス使用許諾契約書 を参照ください。

19.3.2 ソフトウェアのアクティベーションとライセンスの計測

Altova のソフトウェアアクティベーションの一部として、ソフトウェアにより内部ネットワークまたはインターネットへの接続を行い、インストール時、登録時、Altova により使用されるライセンスサーバーの更新やライセンスの正当性を検証することで、ソフトウェアの不正な使用を防ぎ、顧客サービスを向上するため、ライセンスに関する情報を送信することがあります。アクティベーションにより、オペレーティングシステムや IP アドレス、日付/時刻、ソフトウェアのバージョン、コンピュータ名などのライセンスに関する情報が、お使いのコンピューターとAltova ライセンスサーバー間にてやり取りされます。

お使いのAltova 製品にはライセンス計測モジュールが内蔵されており、エンドユーザー使用許諾契約書の意図しない違反を防ぎます。お使いの製品はシングルユーザーまたはマルチユーザーとしてインストールされており、ライセンス計測モジュールにより、ライセンスされている数を超えたユーザーが同時に製品を使用することが無いことが保証されます。

このライセンス計測技術により、ローカルエリア接続 (LAN) において、別々のコンピューター間で動作しているアプリケーションインスタンス間の通信が行われます。

シングルライセンス
ライセンス計測プロセスの一部としてアプリケーションが起動すると、ソフトウェアにより短いデータグラムがブロードキャストにより送信され、同一のネットワークセグメントにある他のコンピューターにてプログラムが動作していないかのチェックが行われます。応答が無い場合は、アプリケーションの他インスタンスから送信される信号に応えるため、ポートが開かれます。

マルチライセンス
同一のLAN 内にて2つ以上のアプリケーションインスタンスが使用された場合、スタートアップ時に、これらインスタンス間において通信が行われます。これらのインスタンス間にてキーコードのやりとりが行われ、購入された数のライセンスを超えてインスタンスが起動しないよう保証することができます。このようなライセンス計測システムはUNIX やデータベース開発ツールにて広く使用されているもので、Altova ユーザーはリーズナブルな価格にて同時使用マルチユーザーライセンスを購入することができます。

弊社はアプリケーションのデザインも行っており、少数の小さなネットワークパケットを送信することで、ネットワークに対する負荷を最小限に抑えております。Altova により使用される2799番 TCP/IP ポートはIANA により公式登録されており(詳細は(IANA Web サイト(http://www.iana.org/) を参照ください)、弊社のライセンス計測モジュールも既にテストされたものです。

ファイヤーウォールを使用している場合、2799番ポートにてAltova 製品が動作しているコンピューター同士が通信しているのに気づかれるかも知れません。その他の手段によりライセンス使用許諾書の内容が守られることを保証できる限り、組織間の異なるグループにおいてこのようなトラフィックをブロックすることは勿論可能です。

お使いのコンピューターがオンラインの場合、ライセンス計測技術とは関係の無い数々の便利な機能を利用することができるようになります。

19.3.3 知的所有権

Altova ソフトウェアならびにAltova により認められたコピーは、Altova ならびにその供給者が知的所有権を有しています。ソフトウェアの構造や構成、コードはAltova ならびにその供給者にとって価値のある企業秘密かつ機密情報となります。このソフトウェアは米国の著作権法により保護されており、国際条項ならびにソフトウェアが使用されている国の法律が適用さ

れます。Altova はこのソフトウェアに関する特許、企業秘密、商標に関する所有権ならびにこれらに限定されないその他の知的財産権を有しており、Altova の所有権には、このソフトウェアならびにそれに付随する印刷物に抱合された画像、写真、アニメーション、映像、音声、音楽、テキスト、そして「アプレット」が含まれます。著作権侵害に関する届け出は、Altova Web Site に記されている Altova 著作権エージェントへお送りください。

Altova ソフトウェアには国際条項により保護されているサードパーティソフトウェアも含まれており、http://www.altova.com/legal_3rdparty.html にて詳細が記されている著作権法なども含め、しかしこれに限定されないかたちで知的所有権により保護されます。

その他全ての名前または商標は、対応する所有者が有するものです。

19.3.4 エンドユーザー使用許諾契約書

THIS IS A LEGAL DOCUMENT - - RETAIN FOR YOUR RECORDS

ALTOVA® END USER LICENSE AGREEMENT

Licensor:
Altova GmbH
Rudolfsplatz 13a/ 9
A- 1010 Wien
Austria

Important - Read Carefully. Notice to User:

This End User License Agreement ("Agreement") is a legal document between you and Altova GmbH ("Altova"). It is important that you read this document before using the Altova- provided software ("Software") and any accompanying documentation, including, without limitation printed materials, 'online' files, or electronic documentation ("Documentation"). By clicking the "I accept" and "Next" buttons below, or by installing, or otherwise using the Software, you agree to be bound by the terms of this Agreement as well as the Altova Privacy Policy ("Privacy Policy") including, without limitation, the warranty disclaimers, limitation of liability, data use and termination provisions below, whether or not you decide to purchase the Software. You agree that this agreement is enforceable like any written agreement negotiated and signed by you. If you do not agree, you are not licensed to use the Software, and you must destroy any downloaded copies of the Software in your possession or control. You may print a copy of this Agreement as part of the installation process at the time of acceptance. Alternatively, a copy of this Agreement may be found at http:/ / www.altova.com/ eula and a copy of the Privacy Policy may be found at http:/ / www.altova.com/ privacy.

1. SOFTWARE LICENSE

(a) License Grant.

(i) Upon your acceptance of this Agreement Altova grants you a non- exclusive, non- transferable (except as provided below), limited license, without the right to grant sublicenses, to install and use a copy of the Software on one compatible personal computer or workstation in the same local area network (LAN) up to the Permitted Number of computers. Subject to the limitations set forth in Section 1(c), you may install and use a copy of the Software on more than one of your compatible personal computers or workstations if you have purchased a Named- User license. Subject to the limitations set forth in Sections 1(d) and 1(e), users may use the software concurrently on a network. The Permitted Number of computers and/ or users and the type of license, e.g. Installed, Named- Users, and Concurrent- User, shall be determined and specified at such time as you elect to purchase the Software. Installed user licenses are intended to be fixed and not concurrent. In other words, you cannot uninstall the Software on one machine in order to reinstall that license to a different machine and then uninstall and reinstall back to the original machine. Installations should be static. Notwithstanding the foregoing, permanent uninstallations and redeployments are acceptable in limited

circumstances such as if an employee leaves the company or the machine is permanently decommissioned. During the evaluation period, hereinafter defined, only a single user may install and use the software on one (1) personal computer or workstation. If you have licensed the Software as part of a suite of Altova software products (collectively, the "Suite") and have not installed each product individually, then the Agreement governs your use of all of the software included in the Suite.

(ii) If you have licensed SchemaAgent, then the terms and conditions of this Agreement apply to your use of the SchemaAgent server software ("SchemaAgent Server") included therein, as applicable, and you are licensed to use SchemaAgent Server solely in connection with your use of Altova Software and solely for the purposes described in the accompanying documentation.

(iii) If you have licensed Software that enables users to generate source code, your license to install and use a copy of the Software as provided herein permits you to generate source code based on (i) Altova Library modules that are included in the Software (such generated code hereinafter referred to as the "Restricted Source Code") and (ii) schemas or mappings that you create or provide (such code as may be generated from your schema or mapping source materials hereinafter referred to as the "Unrestricted Source Code"). In addition to the rights granted herein, Altova grants you a non-exclusive, non-transferable, limited license to compile the complete generated code (comprised of the combination of the Restricted Source Code and the Unrestricted Source Code) into executable object code form, and to use, copy, distribute or license that executable. You may not distribute or redistribute, sublicense, sell, or transfer the Restricted Source Code to a third-party in the un-compiled form unless said third-party already has a license to the Restricted Source Code through their separate agreement with Altova. Notwithstanding anything to the contrary herein, you may not distribute, incorporate or combine with other software, or otherwise use the Altova Library modules or Restricted Source Code, or any Altova intellectual property embodied in or associated with the Altova Library modules or Restricted Source Code, in any manner that would subject the Restricted Source Code to the terms of a copyleft, free software or open source license that would require the Restricted Source Code or Altova Library modules source code to be disclosed in source code form. Notwithstanding anything to the contrary herein, you may not use the Software to develop and distribute other software programs that directly compete with any Altova software or service without prior written permission. Altova reserves all other rights in and to the Software. With respect to the feature(s) of UModel that permit reverse-engineering of your own source code or other source code that you have lawfully obtained, such use by you does not constitute a violation of this Agreement. Except as otherwise expressly permitted in Section 1(j) reverse engineering of the Software is strictly prohibited as further detailed therein.

(iv) In the event Restricted Source Code is incorporated into executable object code form, you will include the following statement in (1) introductory splash screens, or if none, within one or more screens readily accessible by the end-user, and (2) in the electronic and/or hard copy documentation: "Portions of this program were developed using Altova® [name of Altova Software, e.g. MapForce® 2016] and includes libraries owned by Altova GmbH, Copyright © 2007-2016 Altova GmbH (www.altova.com)."

(b) Server Use for Installation and Use of SchemaAgent. You may install one (1) copy of the Software on a computer file server within your internal network solely for the purpose of downloading and installing the Software onto other computers within your internal network up to the Permitted Number of computers in a commercial environment only. If you have licensed SchemaAgent, then you may install SchemaAgent Server on any server computer or workstation and use it in connection with your Software. No other network use is permitted, including without limitation using the Software either directly or through commands, data or instructions from or to a computer not part of your internal network, for Internet or Web-hosting services or by any user not licensed to use this copy of the Software through a valid license from Altova.

(c) Named-Use. If you have licensed the "Named-User" version of the software, you may install the Software on up to five (5) compatible personal computers or workstations of which you are the primary user thereby allowing you to switch from one computer to the other as necessary provided that only one (1) instance of the Software will be used by you as the Named-User at any given time. If you have purchased multiple Named-User licenses, each individual Named-User will receive a separate license key code.

(d) Concurrent Use in Same Local Area Network (LAN). If you have licensed a "Concurrent-User" version of the Software, you may install the Software on any compatible computers in a commercial environment only, up to ten (10) times the Permitted Number of users, provided that only the Permitted Number of users actually use the Software at the same time and further provided that the computers on which the Software is installed are on the same local area network (LAN). The Permitted Number of concurrent users shall be delineated at such time as you elect to purchase the Software licenses. Each separate local area network (LAN) requires its own set of separate Concurrent User Licenses for those wishing to use the Concurrent User versions of the Software in more than one location or on more than one network, all subject to the above Permitted Number limitations and based on the number of users using the Software. If a computer is not on the same local area network (LAN), then a locally installed user license or a license dedicated to concurrent use in a virtual environment is required.

(e) Concurrent Use in Virtual Environment. If you have purchased Concurrent-User Licenses, you may install a copy of the Software on a single host terminal server (Microsoft Terminal Server or Citrix Metaframe), application virtualization server (Microsoft App-V, Citrix XenApp, or VMWare ThinApp) or virtual machine environment within your internal network for the sole and exclusive purpose of permitting individual users within your organization to access and use the Software through a terminal server, application virtualization session, or virtual machine environment from another computer provided that the total number of users that access or use the Software concurrently at any given point in time on such network, virtual machine or terminal server does not exceed the Permitted Number; and provided that the total number of users authorized to use the Software through the terminal server, application virtualization session, or virtual machine environment does not exceed ten (10) times the Permitted Number of users. Key codes for concurrent users cannot be deployed to more than one host terminal server, application virtualization server or virtual machine environment. You must deploy a reliable and accurate means of preventing users from exceeding the Permitted Number of concurrent users. Altova makes no warranties or representations about the performance of Altova software in a terminal server, application virtualization session, or virtual machine environment and the foregoing are expressly excluded from the limited warranty in Section 5 hereof. Technical support is not available with respect to issues arising from use in such environments.

(f) Backup and Archival Copies. You may make one (1) backup and one (1) archival copy of the Software, provided your backup and archival copies are not installed or used on any computer and further provided that all such copies shall bear the original and unmodified copyright, patent and other intellectual property markings that appear on or in the Software. You may not transfer the rights to a backup or archival copy unless you transfer all rights in the Software as provided under Section 3.

(g) Key Codes, Upgrades and Updates. Prior to your purchase and as part of the registration for the thirty (30) day evaluation period, as applicable, you will receive an evaluation key code. You will receive a purchase key code when you elect to purchase the Software from either Altova GmbH or an authorized reseller. The purchase key code will enable you to activate the Software beyond the initial evaluation period. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova. If the Software that you have licensed is an upgrade or an update, then the latest update or upgrade that you download and install replaces all or part of the Software previously licensed. The update or upgrade and the associated license keys does not constitute the granting of a second license to the Software in that you may not use the upgrade or updated copy in addition to the copy of the Software that it is replacing and whose license has terminated.

(h) Title. Title to the Software is not transferred to you. Ownership of all copies of the Software and of copies made by you is vested in Altova, subject to the rights of use granted to you in this Agreement. As between you and Altova, documents, files, stylesheets, generated program code (including the Unrestricted Source Code) and schemas that are authored or created by you via your utilization of the Software, in accordance with its Documentation and the terms of this Agreement, are your property unless they are created using Evaluation Software, as defined in Section 4 of this Agreement, in which case you have only a limited license to use any output that contains generated program code (including Unrestricted Source Code) such as Java, C++, C#, VB.NET or XSLT and associated project files and build scripts, as well as generated XML,

XML Schemas, documentation, UML diagrams, and database structures only for the thirty (30) day evaluation period.

(i) Reverse Engineering. Except and to the limited extent as may be otherwise specifically provided by applicable law in the European Union, you may not reverse engineer, decompile, disassemble or otherwise attempt to discover the source code, underlying ideas, underlying user interface techniques or algorithms of the Software by any means whatsoever, directly or indirectly, or disclose any of the foregoing, except to the extent you may be expressly permitted to decompile under applicable law in the European Union, if it is essential to do so in order to achieve operability of the Software with another software program, and you have first requested Altova to provide the information necessary to achieve such operability and Altova has not made such information available. Altova has the right to impose reasonable conditions and to request a reasonable fee before providing such information. Any information supplied by Altova or obtained by you, as permitted hereunder, may only be used by you for the purpose described herein and may not be disclosed to any third party or used to create any software which is substantially similar to the expression of the Software. Requests for information from users in the European Union with respect to the above should be directed to the Altova Customer Support Department.

(j) Other Restrictions. You may not loan, rent, lease, sublicense, distribute or otherwise transfer all or any portion of the Software to third parties except to the limited extent set forth in Section 3 or as otherwise expressly provided. You may not copy the Software except as expressly set forth above, and any copies that you are permitted to make pursuant to this Agreement must contain the same copyright, patent and other intellectual property markings that appear on or in the Software. You may not modify, adapt or translate the Software. You may not, directly or indirectly, encumber or suffer to exist any lien or security interest on the Software; knowingly take any action that would cause the Software to be placed in the public domain; or use the Software in any computer environment not specified in this Agreement. You may not permit any use of or access to the Software by any third party in connection with a commercial service offering, such as for a cloud-based or web-based SaaS offering.

You will comply with applicable law and Altova's instructions regarding the use of the Software. You agree to notify your employees and agents who may have access to the Software of the restrictions contained in this Agreement and to ensure their compliance with these restrictions.

(k) NO GUARANTEE. THE SOFTWARE IS NEITHER GUARANTEED NOR WARRANTED TO BE ERROR-FREE NOR SHALL ANY LIABILITY BE ASSUMED BY ALTOVA IN THIS RESPECT. NOTWITHSTANDING ANY SUPPORT FOR ANY TECHNICAL STANDARD, THE SOFTWARE IS NOT INTENDED FOR USE IN OR IN CONNECTION WITH, WITHOUT LIMITATION, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION, COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL EQUIPMENT, MEDICAL DEVICES OR LIFE SUPPORT SYSTEMS, MEDICAL OR HEALTH CARE APPLICATIONS, OR OTHER APPLICATIONS WHERE THE FAILURE OF THE SOFTWARE OR ERRORS IN DATA PROCESSING COULD LEAD TO DEATH, PERSONAL INJURY OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR THE ACCURACY AND ADEQUACY OF THE SOFTWARE AND ANY DATA GENERATED OR PROCESSED BY THE SOFTWARE FOR YOUR INTENDED USE AND YOU WILL DEFEND, INDEMNIFY AND HOLD ALTOVA, ITS OFFICERS AND EMPLOYEES HARMLESS FROM ANY THIRD PARTY CLAIMS, DEMANDS, OR SUITS THAT ARE BASED UPON THE ACCURACY AND ADEQUACY OF THE SOFTWARE IN YOUR USE OR ANY DATA GENERATED BY THE SOFTWARE IN YOUR USE.

2. INTELLECTUAL PROPERTY RIGHTS

You acknowledge that the Software and any copies that you are authorized by Altova to make are the intellectual property of and are owned by Altova and its suppliers. The structure, organization and code of the Software are the valuable trade secrets and confidential information of Altova and its suppliers. The Software is protected by copyright, including without limitation by United States Copyright Law, international treaty provisions and applicable laws in the country in which it is being used. You acknowledge that Altova retains the ownership of all patents, copyrights,

trade secrets, trademarks and other intellectual property rights pertaining to the Software, and that Altova's ownership rights extend to any images, photographs, animations, videos, audio, music, text and "applets" incorporated into the Software and all accompanying printed materials. You will take no actions which adversely affect Altova's intellectual property rights in the Software. Trademarks shall be used in accordance with accepted trademark practice, including identification of trademark owners' names. Trademarks may only be used to identify printed output produced by the Software, and such use of any trademark does not give you any right of ownership in that trademark. Altova®, XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™, RaptorXML™, RaptorXML Server™, RaptorXML +XBRL Server™, Powered By RaptorXML™, FlowForce Server™, StyleVision Server™, and MapForce Server™ are trademarks of Altova GmbH. (pending or registered in numerous countries). Unicode and the Unicode Logo are trademarks of Unicode, Inc. Windows, Windows XP, Windows Vista, Windows 7, and Windows 8 are trademarks of Microsoft. W3C, CSS, DOM, MathML, RDF, XHTML, XML and XSL are trademarks (registered in numerous countries) of the World Wide Web Consortium (W3C); marks of the W3C are registered and held by its host institutions, MIT, INRIA and Keio. Except as expressly stated above, this Agreement does not grant you any intellectual property rights in the Software. Notifications of claimed copyright infringement should be sent to Altova's copyright agent as further provided on the Altova Web Site.

3. LIMITED TRANSFER RIGHTS

Notwithstanding the foregoing, you may transfer all your rights to use the Software to another person or legal entity provided that: (a) you also transfer this Agreement, the Software and all other software or hardware bundled or pre-installed with the Software, including all copies, updates and prior versions, and all copies of font software converted into other formats, to such person or entity; (b) you retain no copies, including backups and copies stored on a computer; (c) the receiving party secures a personalized key code from Altova; and (d) the receiving party accepts the terms and conditions of this Agreement and any other terms and conditions upon which you legally purchased a license to the Software. Notwithstanding the foregoing, you may not transfer education, pre-release, or not-for-resale copies of the Software.

4. PRE-RELEASE AND EVALUATION PRODUCT ADDITIONAL TERMS

If the product you have received with this license is pre-commercial release or beta Software ("Pre-release Software"), then this Section applies. In addition, this section applies to all evaluation and/or demonstration copies of Altova software ("Evaluation Software") and continues in effect until you purchase a license. To the extent that any provision in this section is in conflict with any other term or condition in this Agreement, this section shall supersede such other term(s) and condition(s) with respect to the Pre-release and/or Evaluation Software, but only to the extent necessary to resolve the conflict. You acknowledge that the Pre-release Software is a pre-release version, does not represent final product from Altova, and may contain bugs, errors and other problems that could cause system or other failures and data loss. CONSEQUENTLY, THE PRE-RELEASE AND/OR EVALUATION SOFTWARE IS PROVIDED TO YOU **"AS-IS" WITH NO WARRANTIES FOR USE OR PERFORMANCE,** AND ALTOVA DISCLAIMS ANY WARRANTY OR LIABILITY OBLIGATIONS TO YOU OF ANY KIND, WHETHER EXPRESS OR IMPLIED. WHERE LEGALLY LIABILITY CANNOT BE EXCLUDED FOR PRE-RELEASE AND/OR EVALUATION SOFTWARE, BUT IT MAY BE LIMITED, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE SUM OF FIFTY DOLLARS (USD $50) IN TOTAL. If the Evaluation Software has a time-out feature, then the software will cease operation after the conclusion of the designated evaluation period. Upon such expiration date, your license will expire unless otherwise extended. Your license to use any output created with the Evaluation Software that contains generated program code (including Unrestricted Source Code) such as Java, C++, C, VB.NET or XSLT and associated project files and build scripts as well as generated XML, XML Schemas, documentation, UML diagrams, and database structures terminates automatically upon the expiration of the designated evaluation period but the license to use such output is revived upon your purchase of a license for the Software that you evaluated and used to create such output. Access to any files created with the Evaluation Software is entirely at your risk. You acknowledge that Altova has not promised or guaranteed to you that Pre-release Software will be announced or made available to anyone in the future, that Altova has no express or implied obligation to you to announce or introduce the Pre-release Software, and that Altova may not introduce a product similar to or compatible with the Pre-

release Software. Accordingly, you acknowledge that any research or development that you perform regarding the Pre-release Software or any product associated with the Pre-release Software is done entirely at your own risk. During the term of this Agreement, if requested by Altova, you will provide feedback to Altova regarding testing and use of the Pre-release Software, including error or bug reports. If you have been provided the Pre-release Software pursuant to a separate written agreement, your use of the Software is governed by such agreement. You may not sublicense, lease, loan, rent, distribute or otherwise transfer the Pre-release Software. Upon receipt of a later unreleased version of the Pre-release Software or release by Altova of a publicly released commercial version of the Software, whether as a stand-alone product or as part of a larger product, you agree to return or destroy all earlier Pre-release Software received from Altova and to abide by the terms of the license agreement for any such later versions of the Pre-release Software.

5. LIMITED WARRANTY AND LIMITATION OF LIABILITY

(a) Limited Warranty and Customer Remedies. Altova warrants to the person or entity that first purchases a license for use of the Software pursuant to the terms of this Agreement that (i) the Software will perform substantially in accordance with any accompanying Documentation for a period of ninety (90) days from the date of receipt, and (ii) any support services provided by Altova shall be substantially as described in Section 6 of this agreement. Some states and jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you. To the extent allowed by applicable law, implied warranties on the Software, if any, are limited to ninety (90) days. Altova 's and its suppliers ' entire liability and your exclusive remedy shall be, at Altova 's option, either (i) return of the price paid, if any, or (ii) repair or replacement of the Software that does not meet Altova 's Limited Warranty and which is returned to Altova with a copy of your receipt. This Limited Warranty is void if failure of the Software has resulted from accident, abuse, misapplication, abnormal use, Trojan horse, virus, or any other malicious external code. Any replacement Software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. This limited warranty does not apply to Evaluation and/ or Pre-release Software.

(b) No Other Warranties and Disclaimer. THE FOREGOING LIMITED WARRANTY AND REMEDIES STATE THE SOLE AND EXCLUSIVE REMEDIES FOR ALTOVA OR ITS SUPPLIER 'S BREACH OF WARRANTY. ALTOVA AND ITS SUPPLIERS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, AND FOR ANY WARRANTY, CONDITION, REPRESENTATION OR TERM TO THE EXTENT WHICH THE SAME CANNOT OR MAY NOT BE EXCLUDED OR LIMITED BY LAW APPLICABLE TO YOU IN YOUR JURISDICTION, ALTOVA AND ITS SUPPLIERS MAKE NO WARRANTIES, CONDITIONS, REPRESENTATIONS OR TERMS, EXPRESS OR IMPLIED, WHETHER BY STATUTE, COMMON LAW, CUSTOM, USAGE OR OTHERWISE AS TO ANY OTHER MATTERS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, ALTOVA AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, INFORMATIONAL CONTENT OR ACCURACY, QUIET ENJOYMENT, TITLE AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE/ JURISDICTION TO STATE/ JURISDICTION.

(c) Limitation of Liability. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW EVEN IF A REMEDY FAILS ITS ESSENTIAL PURPOSE, IN NO EVENT SHALL ALTOVA OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF ALTOVA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, ALTOVA 'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE PRODUCT. Because some states and jurisdictions do not allow the exclusion or limitation of liability, the above limitation may not apply to you. In such states and jurisdictions,

Altova 's liability shall be limited to the greatest extent permitted by law and the limitations or exclusions of warranties and liability contained herein do not prejudice applicable statutory consumer rights of person acquiring goods otherwise than in the course of business. The disclaimer and limited liability above are fundamental to this Agreement between Altova and you.

(d) Infringement Claims. Altova will indemnify and hold you harmless and will defend or settle any claim, suit or proceeding brought against you by a third party that is based upon a claim that the content contained in the Software infringes a copyright or violates an intellectual or proprietary right protected by United States or European Union law ("Claim"),but only to the extent the Claim arises directly out of the use of the Software and subject to the limitations set forth in Section 5 of this Agreement except as otherwise expressly provided. You must notify Altova in writing of any Claim within ten (10) business days after you first receive notice of the Claim, and you shall provide to Altova at no cost such assistance and cooperation as Altova may reasonably request from time to time in connection with the defense of the Claim. Altova shall have sole control over any Claim (including, without limitation, the selection of counsel and the right to settle on your behalf on any terms Altova deems desirable in the sole exercise of its discretion). You may, at your sole cost, retain separate counsel and participate in the defense or settlement negotiations. Altova shall pay actual damages, costs, and attorney fees awarded against you (or payable by you pursuant to a settlement agreement) in connection with a Claim to the extent such direct damages and costs are not reimbursed to you by insurance or a third party, to an aggregate maximum equal to the purchase price of the Software. If the Software or its use becomes the subject of a Claim or its use is enjoined, or if in the opinion of Altova 's legal counsel the Software is likely to become the subject of a Claim, Altova shall attempt to resolve the Claim by using commercially reasonable efforts to modify the Software or obtain a license to continue using the Software. If in the opinion of Altova 's legal counsel the Claim, the injunction or potential Claim cannot be resolved through reasonable modification or licensing, Altova, at its own election, may terminate this Agreement without penalty, and will refund to you on a pro rata basis any fees paid in advance by you to Altova. THE FOREGOING CONSTITUTES ALTOVA 'S SOLE AND EXCLUSIVE LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT. This indemnity does not apply to situations where the alleged infringement, whether patent or otherwise, is the result of a combination of the Altova software and additional elements supplied by you.

6. SUPPORT AND MAINTENANCE

Altova offers multiple optional "Support & Maintenance Package(s)" ("SMP")for the version of Software product edition that you have licensed, which you may elect to purchase in addition to your Software license. The Support Period, hereinafter defined, covered by such SMP shall be delineated at such time as you elect to purchase a SMP. Your rights with respect to support and maintenance as well as your upgrade eligibility depend on your decision to purchase SMP and the level of SMP that you have purchased:

(a) If you have not purchased SMP, you will receive the Software AS IS and will not receive any maintenance releases or updates. However, Altova, at its option and in its sole discretion on a case by case basis, may decide to offer maintenance releases to you as a courtesy, but these maintenance releases will not include any new features in excess of the feature set at the time of your purchase of the Software. In addition, Altova will provide free technical support to you for thirty (30) days after the date of your purchase (the "Support Period" for the purposes of this paragraph 6(a), and Altova, in its sole discretion on a case by case basis, may also provide free courtesy technical support during your thirty (30) day evaluation period. Technical support is provided via a Web-based support form only, and there is no guaranteed response time.

(b) If you have purchased SMP, then solely for the duration of its delineated Support Period, **you are eligible to receive the version of the Software edition** that you have licensed and all maintenance releases and updates for that edition that are released during your Support Period. For the duration of your SMP 's Support Period, you will also be eligible to receive upgrades to the comparable edition of the next version of the Software that succeeds the Software edition that you have licensed for applicable upgrades released during your Support Period. The specific upgrade edition that you are eligible to receive based on your Support Period is further detailed in the SMP that you have purchased. Software that is introduced as separate product is not included in SMP. Maintenance releases, updates and upgrades may or may not include additional features. In addition, Altova will provide Priority Technical Support to you for

the duration of the Support Period. Priority Technical Support is provided via a Web-based support form only and Altova will make commercially reasonable efforts to respond via e-mail to all requests within forty-eight (48) hours during Altova's business hours (MO-FR, 8am UTC – 10pm UTC, Austrian and US holidays excluded) and to make reasonable efforts to provide work-arounds to errors reported in the Software.

During the Support Period you may also report any Software problem or error to Altova. If Altova determines that a reported reproducible material error in the Software exists and significantly impairs the usability and utility of the Software, Altova agrees to use reasonable commercial efforts to correct or provide a usable work-around solution in an upcoming maintenance release or update, which is made available at certain times at Altova's sole discretion.

If Altova, in its discretion, requests written verification of an error or malfunction discovered by you or requests supporting example files that exhibit the Software problem, you shall promptly provide such verification or files, by email, telecopy, or overnight mail, setting forth in reasonable detail the respects in which the Software fails to perform. You shall use reasonable efforts to cooperate in diagnosis or study of errors. Altova may include error corrections in maintenance releases, updates, or new major releases of the Software. Altova is not obligated to fix errors that are immaterial. Immaterial errors are those that do not significantly impact use of the Software as determined by Altova in its sole discretion. Whether or not you have purchased the Support & Maintenance Package, technical support only covers issues or questions resulting directly out of the operation of the Software and Altova will not provide you with generic consultation, assistance, or advice under any circumstances.

Updating Software may require the updating of software not covered by this Agreement before installation. Updates of the operating system and application software not specifically covered by this Agreement are your responsibility and will not be provided by Altova under this Agreement. Altova's obligations under this Section 6 are contingent upon your proper use of the Software and your compliance with the terms and conditions of this Agreement at all times. Altova shall be under no obligation to provide the above technical support if, in Altova's opinion, the Software has failed due to the following conditions: (i) damage caused by the relocation of the Software to another location or CPU; (ii) alterations, modifications or attempts to change the Software without Altova's written approval; (iii) causes external to the Software, such as natural disasters, the failure or fluctuation of electrical power, or computer equipment failure; (iv) your failure to maintain the Software at Altova's specified release level; or (v) use of the Software with other software without Altova's prior written approval. It will be your sole responsibility to: (i) comply with all Altova-specified operating and troubleshooting procedures and then notify Altova immediately of Software malfunction and provide Altova with complete information thereof; (ii) provide for the security of your confidential information; (iii) establish and maintain backup systems and procedures necessary to reconstruct lost or altered files, data or programs.

7. SOFTWARE ACTIVATION, UPDATES AND LICENSE METERING

(a) License Metering. The Software includes a built-in license metering module that is designed to assist you with monitoring license compliance in small local area networks (LAN). The metering module attempts to communicate with other machines on your local area network (LAN). You permit Altova to use your internal network for license monitoring for this purpose. This license metering module may be used to assist with your license compliance but should not be the sole method. Should your firewall settings block said communications, you must deploy an accurate means of monitoring usage by the end user and preventing users from using the Software more than the Permitted Number.

(b) License Compliance Monitoring. You are required to utilize a process or tool to ensure that the Permitted Number is not exceeded. Without prejudice or waiver of any potential violations of the Agreement, Altova may provide you with additional compliance tools should you be unable to accurately account for license usage within your organization. If provided with such a tool by Altova, you (a) are required to use it in order to comply with the terms of this Agreement and (b) permit Altova to use your internal network for license monitoring and metering and to generate compliance reports that are communicated to Altova from time to time.

(c) Software Activation. **The Software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of**

installation, registration, use, or update to an Altova Master License Server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the Software and to improve customer service. Activation is based on the exchange of license related data between your computer and the Altova Master License Server. You agree that Altova may use these measures and you agree to follow any applicable requirements. You further agree that use of license key codes that are not or were not generated by Altova and lawfully obtained from Altova, or an authorized reseller as part of an effort to activate or use the Software violates Altova's intellectual property rights as well as the terms of this Agreement. You agree that efforts to circumvent or disable Altova's copyright protection mechanisms, the license management mechanism, or the Altova Master License Server violate Altova's intellectual property rights as well as the terms of this Agreement. Altova expressly reserves the rights to seek all available legal and equitable remedies to prevent such actions and to recover lost profits, damages and costs.

(d) LiveUpdate. Altova provides a new LiveUpdate notification service to you, which is free of charge. Altova may use your internal network and Internet connection for the purpose of transmitting license-related data to an Altova-operated LiveUpdate server to validate your license at appropriate intervals and determine if there is any update available for you.

(e) Use of Data. The terms and conditions of the Privacy Policy are set out in full at http://www.altova.com/privacy and are incorporated by reference into this Agreement. By your acceptance of the terms of this Agreement and/or use of the Software, you authorize the collection, use and disclosure of information collected by Altova for the purposes provided for in this Agreement and/or the Privacy Policy. Altova has the right in its sole discretion to amend this provision of the Agreement and/or Privacy Policy at any time. You are encouraged to review the terms of the Privacy Policy as posted on the Altova Web site from time to time.

(f) Audit Rights. You agree that Altova may audit your use of the Software for compliance with the terms of this Agreement at any time, upon reasonable notice. In the event that such audit reveals any use of the Software by you other than in full compliance with the terms of this Agreement, you shall reimburse Altova for all reasonable expenses related to such audit in addition to any other liabilities you may incur as a result of such non-compliance.

(g) Notice to European Users. Please note that the information as described in paragraph 7(d) above may be transferred outside of the European Economic Area, for purposes of processing, analysis, and review, by Altova, Inc., a company located in Beverly, Massachusetts, U.S.A., or its subsidiaries or Altova's subsidiaries or divisions, or authorized partners, located worldwide. You are advised that the United States uses a sectoral model of privacy protection that relies on a mix of legislation, governmental regulation, and self-regulation. You are further advised that the Council of the European Union has found that this model does not provide "adequate" privacy protections as contemplated by Article 25 of the European Union's Data Directive. (Directive 95/46/EC, 1995 O.J. (L 281) 31). Article 26 of the European Union's Data Directive allows for transfer of personal data from the European Union to a third country if the individual has unambiguously given his consent to the transfer of personal information, regardless of the third country's level of protection. By agreeing to this Agreement, you consent to the transfer of all such information to the United States and the processing of that information as described in this Agreement and the Privacy Policy.

8. TERM AND TERMINATION

This Agreement may be terminated (a) by your giving Altova written notice of termination; (b) by Altova, at its option, giving you written notice of termination if you commit a breach of this Agreement and fail to cure such breach within ten (10) days after notice from Altova; or (c) at the request of an authorized Altova reseller in the event that you fail to make your license payment or other monies due and payable. In addition the Agreement governing your use of a previous version of the Software that you have upgraded or updated is terminated upon your acceptance of the terms and conditions of the Agreement accompanying such upgrade or update. Upon any termination of the Agreement, you must cease all use of the Software that this Agreement governs, destroy all copies then in your possession or control and take such other actions as Altova may reasonably request to ensure that no copies of the Software remain in

your possession or control. The terms and conditions set forth in Sections 1(h), 1(i), 1(j), 1(k), 1(l), 2, 5, 7, 9, 10, 11, and 11 survive termination as applicable.

9. RESTRICTED RIGHTS NOTICE AND EXPORT RESTRICTIONS

The Software was developed entirely at private expense and is commercial computer software provided with **RESTRICTED RIGHTS**. Use, duplication or disclosure by the U.S. Government or a U.S. Government contractor or subcontractor is subject to the restrictions set forth in this Agreement and as provided in FAR 12.211 and 12.212 (48 C.F.R. § 12.211 and 12.212) or DFARS 227. 7202 (48 C.F.R. § 227- 7202) as applicable. Consistent with the above as applicable, Commercial Computer Software and Commercial Computer Documentation licensed to U.S. government end users only as commercial items and only with those rights as are granted to all other end users under the terms and conditions set forth in this Agreement. Manufacturer is Altova GmbH, Rudolfsplatz 13a/ 9, A- 1010 Vienna, Austria/ EU. You may not use or otherwise export or re- export the Software or Documentation except as authorized by United States law and the laws of the jurisdiction in which the Software was obtained. In particular, but without limitation, the Software or Documentation may not be exported or re- exported (i) into (or to a national or resident of) any U.S. embargoed country or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals or the U.S. Department of Commerce's Table of Denial Orders. By using the Software, you represent and warrant that you are not located in, under control of, or a national or resident of any such country or on any such list.

10. U.S. GOVERNMENT ENTITIES

Notwithstanding the foregoing, if you are an agency, instrumentality or department of the federal government of the United States, then this Agreement shall be governed in accordance with the laws of the United States of America, and in the absence of applicable federal law, the laws of the Commonwealth of Massachusetts will apply. Further, and notwithstanding anything to the contrary in this Agreement (including but not limited to Section 5 (Indemnification)), all claims, demands, complaints and disputes will be subject to the Contract Disputes Act (41 U.S.C. § § 7101 *et seq.*), the Tucker Act (28 U.S.C. § 1346(a) and § 1491), or the Federal Tort Claims Act (28 U.S.C. § § 1346(b), 2401- 2402, 2671- 2672, 2674- 2680), FAR 1.601(a) and 43.102 (Contract Modifications); FAR 12.302(b), as applicable, or other applicable governing authority. For the avoidance of doubt, if you are an agency, instrumentality, or department of the federal, state or local government of the U.S. or a U.S. public and accredited educational institution, then your indemnification obligations are only applicable to the extent they would not cause you to violate any applicable law (e.g., the Anti- Deficiency Act), and you have any legally required authorization or authorizing statute.

11. THIRD PARTY SOFTWARE

The Software may contain third party software which requires notices and/ or additional terms and conditions. Such required third party software notices and/ or additional terms and conditions are located at our Website at http:/ / www.altova.com/ legal_3rdparty.html and are made a part of and incorporated by reference into this Agreement. By accepting this Agreement, you are also accepting the additional terms and conditions, if any, set forth therein.

12. JURISDICTION, CHOICE OF LAW, AND VENUE

If you are located in the European Union and are using the Software in the European Union and not in the United States, then this Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht, Wien (Commercial Court, Vienna) in connection with any such dispute or claim.

If you are located in the United States or are using the Software in the United States then this Agreement will be governed by and construed in accordance with the laws of the Commonwealth of Massachusetts, USA (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction

for any claim or dispute with Altova or relating in any way to your use of the Software resides in the federal or state courts of the Commonwealth of Massachusetts and you further agree and expressly consent to the exercise of personal jurisdiction in the federal or state courts of the Commonwealth of Massachusetts in connection with any such dispute or claim.

If you are located outside of the European Union or the United States and are not using the Software in the United States, then this Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht Wien (Commercial Court, Vienna) in connection with any such dispute or claim. This Agreement will not be governed by the conflict of law rules of any jurisdiction or the United Nations Convention on Contracts for the International Sale of Goods, the application of which is expressly excluded.

13. TRANSLATIONS

Where Altova has provided you with a foreign translation of the English language version, you agree that the translation is provided for your convenience only and that the English language version will control. If there is any contradiction between the English language version and a translation, then the English language version shall take precedence.

14. GENERAL PROVISIONS

This Agreement contains the entire agreement and understanding of the parties with respect to the subject matter hereof, and supersedes all prior written and oral understandings of the parties with respect to the subject matter hereof. Any notice or other communication given under this Agreement shall be in writing and shall have been properly given by either of us to the other if sent by certified or registered mail, return receipt requested, or by overnight courier to the address shown on Altova 's Web site for Altova and the address shown in Altova 's records for you, or such other address as the parties may designate by notice given in the manner set forth above. This Agreement will bind and inure to the benefit of the parties and our respective heirs, personal and legal representatives, affiliates, successors and permitted assigns. The failure of either of us at any time to require performance of any provision hereof shall in no manner affect such party 's right at a later time to enforce the same or any other term of this Agreement. This Agreement may be amended only by a document in writing signed by both of us. In the event of a breach or threatened breach of this Agreement by either party, the other shall have all applicable equitable as well as legal remedies. Each party is duly authorized and empowered to enter into and perform this Agreement. If, for any reason, any provision of this Agreement is held invalid or otherwise unenforceable, such invalidity or unenforceability shall not affect the remainder of this Agreement, and this Agreement shall continue in full force and effect to the fullest extent allowed by law. The parties knowingly and expressly consent to the foregoing terms and conditions.

Last updated: 2015/ 09/ 03

Chapter 20

用語

20 用語

この用語のセクションはMapForce に関連する用語のリストを含みます。

(C) 2017 Altova GmbH

20.1 C

コンポーネント

MapForce 内では、"コンポーネント"という用語はデータの構造 (スキーマ) またはデータがどのように変換 (関数) されるかを視覚的に表します。コンポーネントはすべてのマッピングを構築する中心的な役割を果たします。以下は、MapForce コンポーネントの例です :

- 定数
- データベース
- フィルター
- 条件
- フラットファイル (CSV、固定長と他のテキストファイル)
- 関数コンポーネント
- EDI ドキュメント (UN/EDIFACT、ANSI X12、HL7)
- Excel 2007+ ファイル
- 単純型 入力コンポーネント
- 単純型 出力コンポーネント
- 変数
- XBRL ドキュメント
- スキーマおよびDTD

接続

接続とは2つのコネクタ間に描くことができる線です。接続を描くことにより、MapForce にデータを特定の方法で変換するように命令します。(例 :XML ドキュメントからデータを読み込み、他のXML ドキュメントに書き込みます)。

コネクタ

コンポーネントの左右に表示される小さな三角形です。コンポーネントの左に表示されるコネクタはそのコンポーネントのエントリポイントを表します。コンポーネントの右に表示されるコネクタはそのコンポーネントからのデータの終了ポイントを表します。

20.2 F

固定長フィールド (FLF)

データが習慣的に固定値を持つフィールドに分割される共通のテキストフォーマット。 (例 :行の最初の 5文字がトランザクションID を表し、次の 20文字がトランザクションの詳細を表すなど)。

FlexText

FlexText とは MapForce によりサポートされる他のフォーマットに変換するには複雑な非標準またはレガシーテキストファイルからのデータを変換するMapForce Enterprise Edition 内のモジュールです。

20.3 I

入力コンポーネント

入力コンポーネントは、単純型の値をマッピングにパスすることができるMapForce コンポーネントです。入力コンポーネントは通常ファイル名またはその他の文字列の値をランタイムにマッピングにパスするために使用されます。入力コンポーネントとソースコンポーネントを混同しないようご注意ください。

20.4 J

ジョインコンポーネント

ジョインコンポーネントは、カスタム定義条件をベースにしたマッピングの2つの構造をジョインすることができる MapForce コンポーネントです。ジョイン機能は、共通のフィールド (ID などの) を共有する2つのソース構造からデータを組み合わせる際に役にたちます。データベースの場合、ジョインコンポーネントを使用して、マッピングの実行の速度をスピードアップする、SQL モード内の2つのテーブルをジョインするために使用されます。データベーステーブルと (異なるデータベースのテーブル間のジョインを含む) 他のコンポーネント型間のジョインも可能です。 詳細に関しては、以下を参照してください:
データのジョイン。

20.5 M

MapForce

MapForce は、プログラムコードを作成することなく視覚的なドラッグアンドドロップを使用するグラフィカルなユーザーインターフェイスを用いデータを異なるデータ間、スキーマ間で変換するWindows ベースの多目的のIDE (統合された開発環境 統合開発環境)です。また、事実上、MapForce は時際のデータ変換 (またデータマッピング)を行うプログラムコードを生成します。プログラムコードを生成しない場合、 (MapForce Professional またはEnterprise Editionsで使用することのできる)MapForce 内蔵の変換言語 を実行します。

マッピング

MapForce マッピングデザイン (または、略して "マッピング")は、データがどのように1つのフォーマットから他のフォーマットに変換されるかの視覚的な表現です。マッピングは、データ変換を行うためのMapForce マッピングエリア内で1つのスキーマから他のスキーマに追加するコンポーネントから構成されています。 (例 :XML ドキュメントを1つのスキーマから他のスキーマに変換) 。有効なマッピングは、1つまたは複数のターゲットコンポーネントに接続されている、1つまたは複数の複数のソースコンポーネントから構成されています。マッピングを実行して、結果を直接 MapForce 内で確認することができます。コードを生成し、外部で実行することも可能です。MapForce 実行可能ファイルにマッピングをコンパイルし、MapForce Server またはFlowForce Server を使用してマッピングの実行を自動化することもできます。 MapForce は、マッピングを .mfd の拡張子を持つファイルとして保存します。

MFF

MapForce 関数ファイルのファイル名の拡張子

MFD

MapForce デザインドキュメントのファイル名の拡張子 (マッピング)

MFP

MapForce プロジェクトファイルのファイル名の拡張子

MFT

MapForce FlextText テンプレートドキュメントのファイル名の拡張子

20.6 O

出力コンポーネント

出力コンポーネント (または "単純型出力 ") とは、マッピングから文字列の値を返すことを有効化するMapForce コンポーネントです。出力 コンポーネントは、1つのターゲットコンポーネント型のみをあらわしますが、後者と混同しないように注意してください。

20.7 P

parent-context

`min`、`max`、`avg`、`count` などのMapForce 一部のコア集計関数 **parent-context** 任意の引数です。複数の階層的シーケンスを持つソースコンポーネントでは、親コンテキストは、ノードのセットがどの関数上で操作されるかを決定します。

20.8 S

ソース コンポーネント

ソースコンポーネントとは、MapForce がデータを読み込む元のコンポーネントです。マッピングを実行すると、ソースコンポーネントのコネクタにより与えられたデータを読み込み、必要とされる型に変換し、ターゲットコンポーネントのコネクタに送信します。

20.9 T

ターゲットコンポーネント

ターゲットコンポーネントとMapForce はがデータを書き込むコンポーネントです。マッピング を実行すると、ターゲットコンポーネントは、外部プログラム内で更に処理するためにMapForce にファイル (または複数のファイル)の生成、または結果を文字列の値として出力するようご命令します。ターゲットコンポーネントは、ソース コンポーネントの逆です。

Index

.

.NET 拡張関数 ,
XSLT と XQuery, 1430
インスタンスメソッド、インスタンスフィールド ,1433
コンストラクター ,1432
データ型変換、 .NET から XPath/ XQuery へ ,1435
データ型変換、XPath/ XQuery から .NET へ ,1434
概要 ,1430
静的メソッド、静的フィールド ,1433
.NET 内の XSLT と XQuery のための拡張機能 ,
.NET 拡張関数を参照する ,1430

1

1回分割する ,
using the "fixed length" mode, 542
区切られた (フロート) モードの使用 ,543
区切られた (行ベース) モードの使用 ,543,545

2

2017, 1204

A

A から Z,
並べ替えコンポーネント ,170
aar ファイル ,
のための Axis2axis アーカイブ ,877
abs,
MapForce 関数として (lang | math 関数),843
MapForce 関数として (xpath2 | numeric 関数),859
acos,
MapForce 関数として (lang | math 関数),843
Active, 1252
ActiveDocument, 1214, 1247
ActiveX,
integration at application level, 1295
integration at document level, 1297
integration prerequisites, 1291
ActiveX controls,
adding to the Visual Studio Toolbox, 1293
ADO,
データ接続インターフェイスとして ,300
接続のセットアップ ,305
ADO.NET,
接続のセットアップ ,309
age,
MapForce 関数として (lang | datetime 関数),829
Altova XML パーサー ,
について ,1438
Altova エンジン ,
Altova 製品内で ,1438
Altova 拡張関数 ,
チャート関数 (チャート関数を参照),1364
ANSI X12,
メッセージ構造のカスタマイズ ,636
Ant,
Building an Eclipse project with, 1069
Java Web サービスのコンパイル ,877
setting the environment variables for,, 1166
Any,
xs:any, 283
API,
documentation, 1186
overview, 1187
Application, 1212
for Documents, 1247
Application object, 1189, 1213
AppOutputLine, 1220
AppOutputLines, 1225
AppOutputLineSymbol, 1226
ASC X12, 565, 575
asin,
MapForce 関数として (lang | math 関数),843
atan,
MapForce 関数として (lang | math 関数),843
ATTLIST,
DTD 名前空間 URI, 275
auto- format,
MapForce 関数として (edifact 関数),827
auto- number,
MapForce 関数として (コア | generator 関数),796
avg,
MapForce 関数として (コア | aggregate 関数),782

B

Background,
with gradient, 1266
Base package name,
for Java, 1243
base- uri,
MapForce 関数として (xpath2 | accessors ライブラリ),853
Bool,
false の場合の出力 ,734
boolean,
MapForce 関数として (コア | conversion 関数),785
build.xml,
enabling the zip64mode in„ 1166

C

C#,
C# Web サービスの作成 ,883
code, 1054
code generation, 1240
code generation settings, 1167
enumeration, 1286
error handling, 1189
generating mapping code in, 1063
integrate generated code, 1080
integration of MapForce, 1301
options, 1262, 1264
C++,
code, 1054
code generation, 1241
code generation settings, 1167
enumeration, 1285
error handling, 1189
generating mapping code in, 1059
integrate generated code, 1082
options, 1262, 1263
capitalize,
MapForce 関数として (lang | string 関数),846
CDATA, 282
ceiling,
MapForce 関数として (コア | math 関数),801
char- from- code,
MapForce 関数として (コア | string 関数),818
Class ID,
in MapForce integration, 1307
Close, 1239
project, 1269
Code,
built in types, 1179
integrating MapForce code, 1078
SPL, 1168
code generation, 1248, 1250
C#, 1240
C++, 1241
enumerations, 1282, 1285, 1286, 1287
Java, 1241
options, 1245
options for, 1262, 1263, 1264, 1265
sample, 1202
XSLT, 1242
Code Generator, 1054
code- from- char,
MapForce 関数として (コア | string 関数),818
Code- generation,
options for, 1218, 1266
columnname- to- index,
MapForce 関数として ,851
COM API,
documentation, 1186
Component, 1227
Components, 1234
concat,
(関数として)サンプルの使用方法 ,506
MapForce 関数として (コア | string 関数),818
Connection, 1235
contains,
MapForce 関数として (コア | string 関数),819
convert- to- utc,
MapForce 関数として (lang | datetime 関数),830
cos,
MapForce 関数として (lang | math 関数),844
Count, 1225, 1235, 1247, 1261
MapForce 関数として (コア | aggregate 関数),782
count- substring,
MapForce 関数として (lang | string 関数),847
create- guid,
MapForce 関数 (lang | generator 関数),842
CSV,
マッピングソースとして ,495
階層の作成 - キー ,499
作成 複数の 行 ,498
CSV として保管 区切られた),

CSV として保管 区切られた),
FlexText 関数として ,550
CSV ファイル ,
ソース コンポーネントとして ,502
ターゲットコンポーネントとして ,502
のエンコードを設定 ,502
プレビュー データ,502
追加または削除フィールド,502
current,
MapForce 関数として (xslt | xslt 関数ライブラリ),863
current- date,
MapForce 関数として (xpath2 | context 関数),855
current- dateTime,
MapForce 関数として (xpath2 | context 関数),855
current- time,
MapForce 関数として (xpath2 | context 関数),855

D

Datapoint, 1236
date- from- datetime,
MapForce 関数として (lang | datetime 関数),830
datetime- add,
MapForce 関数として (lang | datetime 関数),830
datetime- diff,
MapForce 関数として (lang | datetime 関数),832
datetime- from- date- and- time,
MapForce 関数として (lang | datetime 関数),832
datetime- from- parts,
MapForce 関数として (lang | datetime 関数),833
datetime- to- xlsx,
MapForce 関数として ,852
date- to- xlsx,
MapForce 関数として ,852
day- from- datetime,
MapForce 関数として (lang | datetime 関数),834
day- from- duration,
MapForce 関数として (lang | datetime 関数),835
DB,
ORDER BY, 437
default- collation,
MapForce 関数として (xpath2 | context 関数),855
degrees,
MapForce 関数として (lang | math 関数),844
Digital certificates,
exporting from Windows, 930
in MapForce mappings, 921
managing on Windows, 929
transferring to Linux, 933
transferring to Mac, 934
trusting on Linux, 923
trusting on Mac, 926
trusting on Windows, 927
distinct- values,
MapForce 関数として (コア | sequence 関数),805
divide,
MapForce 関数として (コア | math 関数),801
divide- integer,
MapForce 関数として (lang | math 関数),844
Document, 1237
closing, 1239
creating new, 1217, 1248
filename, 1244
MapForce 関数として (xslt | xslt 関数ライブラリ),863
on closing, 1239
on opening, 1213
opening, 1217, 1248
path and name of, 1240
path to, 1245
retrieving active document, 1247
save, 1245, 1246
save as, 1246
Document- level,
examples of integration of XMLSpy, 1301
Documents, 1214, 1246
retrieving, 1247
total number in collection, 1247
DOM type,
enumerations for C++, 1285
for C++, 1262
DoTransform.bat,
RaptorXML Server を使用して実行 ,946
DTD,
ソースとターゲット,275
duration- add,
MapForce 関数として (lang | datetime 関数),835
duration- from- parts,
MapForce 関数として (lang | datetime 関数),835
duration- subtract,
MapForce 関数として (lang | datetime 関数),836

E

Eclipse,

Eclipse,
generating mapping code for, 1066
Eclipse のための MapForce プラグイン,
creating a MapForce/ Eclipse プロジェクト,1017
Eclipse プロジェクトにマッピングをインポートする,1020
MapForce パースペクティブに切り替える,1011
インストール,1007
について,1006,1011,1017
マッピングとプロジェクトとの作業,1017
拡張機能,1024
拡張点,1024
関数共通のメニューと関数の評価,1014
自動コードの生成の設定,1022
新規マッピングの作成,1019
EDI,
ASC X12, 575
IATA PADIS, 586
X12 HIPAA, 579
セパレーター,567
検証のカスタマイズ,642
EDI ファイル,
マッピングコンポーネントととして,567
EDIFACT, 565
XML にマッピング,608
マージされたエントリ,624
用語,567
Edition, 1215
element- available,
MapForce 関数として (xslt | xslt 関数ライブラリ),863
empty,
MapForce 関数として (lang | string 関数),847
Encoding,
default for output files, 1264
Enumerations, 1282, 1283, 1284, 1285, 1286
for MapForce View, 1287
in MapForceControl, 1353
Environment variables,
ANT_OPS, 1166
equal,
MapForce 関数として (ロア | logical 関数),799
equal- or- greater,
MapForce 関数として (ロア | logical 関数),799
equal- or- less,
MapForce 関数として (ロア | logical 関数),799
Error handling,
general description, 1189
ErrorMarkers, 1248, 1250
Events,
of Document, 1238
Events for Project, 1267
Excel 2007+,
RSS フィードからのマッピング,914
XML にマッピングする,674
XML ファイルから複数のファイルを生成する,144
セルの範囲からデータのマッピングを読み込む,668
セルの範囲にデータのマッピングを書き込む,668
へマッピング データ,205
マッピング from データベース to, 676
マッピング コンポーネントからセルの範囲を選択する,669
マッピング コンポーネントとして,664
マッピング コンポーネントとして追加,666
マッピング ソース またはターゲットとして,661
Excel 2007+ コンポーネント設定,
変更,672
Excel 2007+ ワークシート,
からのデータの読み込み,667
へのデータの書き込み,667
exists,
MapForce 関数として (ロア | sequence 関数),806
exp,
MapForce 関数として (lang | math 関数),844

F

false,
MapForce 関数として (xpath2 | boolean 関数),854
find- substring,
MapForce 関数として (lang | string 関数),847
Firebird,
ODBC を使用しての接続,326
first- items,
MapForce 関数として (ロア | sequence 関数),807
FlexText,
インターフェイス,520
コンテナー,520
コンポーネント設置の変更,532
ターゲットコンポーネントとして,534
チュートリアル,522
の定義,1456
概要の,520
FlexText 関数,
CSV として保管 (区切られた),550
FLF として保管 (固定長),555
Split once, 541
Store value, 557
ノード,549

FlexText 関数 ,
繰り返される分割 ,535
切り替え ,546
無視する ,550
FLF,
の定義 ,1456
FLF として保管 固定長),
FlexText 関数として ,555
floor,
MapForce 関数として (コア | math 関数),802
format- date,
MapForce 関数として (コア | conversion 関数),786
format- dateTime,
MapForce 関数として (コア | conversion 関数),786
format- guid- string,
MapForce 関数として (lang | string 関数),847
format- number,
MapForce 関数として (コア | conversion 関数),789
format- time,
MapForce 関数として (コア | conversion 関数),790
FullName, 1240, 1270
function- available,
MapForce 関数として (xslt | xslt 関数ライブラリ),864

G

Generate,
code from schema, 1054
generate- id,
MapForce 関数として (xslt | xslt 関数ライブラリ),864
generate- sequence,
MapForce 関数として (コア | sequence 関数),807
get- fileext,
MapForce 関数として (コア | file path 関数),794
get- folder,
MapForce 関数として (コア | file path 関数),794
GetRootDatapoint, 1283
Global objects,
in SPL, 1171
Globalresource.xml,
リソース 定義 ,957
Gradients,
in background, 1266
greater,
MapForce 関数として (コア | logical 関数),799
group- adjacent,
MapForce 関数として (コア | sequence 関数),807
group- by,
MapForce 関数として (コア | sequence 関数),809
group- ending- with,
MapForce 関数として (コア | sequence 関数),810
group- into- blocks,
MapForce 関数として (コア | sequence 関数),811
group- starting- with,
MapForce 関数として (コア | sequence 関数),811

H

Health Level 7,
サンプル ,714
HighlightMyConnections, 1253
HighlightMyConnectionsRecursively, 1253
HIPAA,
X12, 579
X12 999 Implementation Acknowledgement の生成 ,578
HIPAA X12, 565
HL7, 565
HL7 2.6 to 3.x,
サンプル ,714
hour- from- datetime,
MapForce 関数として (lang | datetime 関数),837
hour- from- duration,
MapForce 関数として (lang | datetime 関数),837
HRESULT,
and error handling, 1189
HTML,
integration of MapForce, 1309
マッピング ドキュメント ,981
マッピングの出力のプレビュー ,977
HTML example,
of MapForceControl integration, 1307, 1308, 1309
HTTPS,
calling Web services through, 921

I

IATA PADIS, 565, 586
IBM DB2,
ODBC を使用しての接続 ,329
XML 型フィールドから読み込む ,451
XML 型フィールドへ書き込む ,451
IBM DB2 for i,

IBM DB2 for i,
ODBC を使用しての接続 ,333
IBM Informix,
JDBC を使用しての接続 ,336
If- Else 条件 ,
マッピングに追加 ,175
implicit- timezone,
MapForce 関数として (xpath2 | context 関数),855
index- to- columnname,
MapForce 関数として ,852
InsertXMLFile, 1254
InsertXMLSchema, 1254
InsertXMLSchemaWithSample, 1255
Integrate,
into C#, 1080
into C++, 1082
into Java, 1079
Integrate MapForce code, 1078
Integrating,
MapForce in applications, 1290
Introduction,
code generator, 1055
is- not- null,
MapForce 関数として (db 関数),825
is- null,
MapForce 関数として (db 関数),826
is- xsi- nil,
MapForce 関数として (コア | node 関数),803
Item, 1225, 1235, 1247, 1261
item- at,
MapForce 関数として (コア | sequence 関数),812
items- from- till,
MapForce 関数として (コア | sequence 関数),812

J

Java, 1314
avoiding exceptions in generated code, 1166
code, 1054
code generation, 1241
generating mapping code in, 1066
integrate generated code, 1079
options, 1243, 1262, 1265
Web サービスの作成 ,877
Java 拡張関数 ,
XSLT と XQuery, 1422
インスタンスメソッド、インスタンスフィールド ,1428
コンストラクター ,1427
データ型変換、Java から XPath/ XQuery へ ,1430
データ型変換、XPath/ XQuery から Java へ ,1429
ユーザー定義の JAR ファイル ,1426
ユーザー定義のクラスファイル ,1423
概要 ,1422
静的メソッド、静的フィールド ,1427
JavaScript,
error handling, 1189
JDBC,
handling references in generated code, 1068
Mac OS X Yosemite 上での Oracle 接続のセットアップ ,357
データ接続インターフェイスとして ,300
接続のセットアップ (Linux), 356
接続のセットアップ (Mac OS), 356
接続の設定 (Windows), 317
JScript,
code- generation sample, 1202
JSON,
Web サービスからのマッピング ,911
マッピング サンプル ,656
追加プロパティからのマッピング ,656
JSON ファイル ,
ソーまたはターゲット コンポーネントとして ,650

L

last,
MapForce 関数として (xpath2 | context 関数),855
last- items,
MapForce 関数として (コア | sequence 関数),812
leapyear,
MapForce 関数として (lang | datetime 関数),837
left,
MapForce 関数として (lang | string 関数),847
left- trim,
MapForce 関数として (lang | string 関数),848
less,
MapForce 関数として (コア | logical 関数),799
Library, 1180
Library type,
enumerations for C++, 1285
for C++, 1263
Linux,
executing mappings with Web service calls through HTTPS, 933
transferring client certificates to, 933

Linux,
trusting server certificates on, 923
サーバー実行ファイルをデプロイ ,355
サポートされるデータベース ,355
データベース接続のセットアップ ,355
local- name- from- QName,
MapForce 関数として (ang | QName 関数),779
log,
MapForce 関数として (ang | math 関数),844
log10,
MapForce 関数として (ang | math 関数),844
logical- and,
MapForce 関数として (ロア | logical 関数),799
logical- not,
MapForce 関数として (ロア | logical 関数),800
logical- or,
MapForce 関数として (ロア | logical 関数),800
logical- xor,
MapForce 関数として (ang | logical 関数),842
Logo,
display on startup, 1266
option for printing, 1265
lowercase,
MapForce 関数として (ang | string 関数),848

M

Mac,
executing mappings with Web service calls through HTTPS, 934
transferring client certificates to, 934
trusting server certificates on, 926
Mac OS,
サーバー実行ファイルをデプロイ ,355
サポートされるデータベース ,355
データベース接続のセットアップ ,355
main- mfd- filepath,
MapForce 関数として (ロア | file path 関数),794
MapForce,
API, 1186
integration, 1290
parent, 1255
概要 ,13
基本概念 ,18
MapForce API, 1186
overview, 1187
MapForce API Type Library, 1212
MapForce integration,
example of, 1307, 1308, 1309
MapForce plug- in for Visual Studio,
について ,998
MapForce samples,
location on disk, 28
MapForce Server,
から例外をスロー ,199
のためにマッピングをコンパイルする ,940
マッピングの自動化 ,939
MapForce view,
enumerations for, 1287
MapForceCommand,
in MapForceControl, 1334
MapForceCommands,
in MapForceControl, 1336
MapForceControl, 1337
documentation of, 1290
example of integration at application level, 1307, 1308, 1309
examples of integration at document level, 1301
integration using C#, 1301
integration using HTML, 1309
integration using Visual Basic, 1322
object reference, 1334
MapForceControlDocument, 1344
MapForceControlPlaceHolder, 1350
MapForceView, 1244, 1252
application, 1253
Mapping,
inserting XML file, 1254
inserting XML Schema file, 1254
match- pattern,
MapForce 関数として (ang | string 関数),848
max,
MapForce 関数として (ang | math 関数),845
MapForce 関数として (ロア | aggregate 関数),782
max- string,
MapForce 関数として (ロア | aggregate 関数),783
MERGE,
MapForce- により生成された SQL ステートメントとして , 400
MapForce- により生成された SQL 内のステートメントとして ,395
Method names in generating code,
reserving, 1166
MFC support,
fo C++, 1263
mfd,
ファイル拡張子として ,1459
mfd- filepath,

mfd- filepath,
MapForce 関数として (コア | file path 関数),795
mff, 763
mff.xsd ファイル ,763
ファイル拡張子として ,1459
ライブラリファイル ,763
mff ファイル ,
構成 ,763
mfp,
ファイル拡張子として ,1459
mft,
ファイル拡張子として ,1459
Microsoft Access,
ADO を介しての接続 ,305
ADO を使用しての接続 ,337
Microsoft Excel,
"Excel 2007+" を参照 ,661
Microsoft SharePoint Server,
ファイルをコンポーネントとして追加 ,64
Microsoft SQL Server,
ADO を使用しての接続 ,338
ODBC を使用しての接続 ,341
millisecond- from- datetime,
MapForce 関数として (lang | datetime 関数),838
millisecond- from- duration,
MapForce 関数として (lang | datetime 関数),838
min,
MapForce 関数として (lang | math 関数),845
MapForce 関数として (コア | aggregate 関数),783
min- string,
MapForce 関数として (コア | aggregate 関数),784
minute- from- datetime,
MapForce 関数として (lang | datetime 関数),838
minute- from- duration,
MapForce 関数として (lang | datetime 関数),839
modulus,
MapForce 関数として (コア | math 関数),802
month- from- datetime,
MapForce 関数として (lang | datetime 関数),839
month- from- duration,
MapForce 関数として (lang | datetime 関数),839
MSXML library,
generating code for, 1167
msxsl:script, 1435
multiply,
MapForce 関数として (コア | math 関数),802
MySQL,
ODBC を使用しての接続 ,342

N

Name, 1217, 1244, 1272
namespace- uri- from- QName,
MapForce 関数として (lang | QName 関数),780
negative,
MapForce 関数として (lang | logical 関数),842
NewDocument, 1217, 1248
NewProject, 1217
nillable,
XML スキーマ内の属性として ,278
node- name,
MapForce 関数として (コア | node 関数),803
MapForce 関数として (xpath2 | accessors ライブラリ),853
node- name 関数 ,215
代替の使用方法 ,215
normalize- space,
MapForce 関数として (コア | string 関数),819
not- equal,
MapForce 関数として (コア | logical 関数),800
not- exists,
MapForce 関数として (コア | sequence 関数),812
now,
MapForce 関数として (lang | datetime 関数),839
NULL,
データベースコンポーネントの処理 ,379
データベースマッピング内での扱い ,417
number,
MapForce 関数として (コア | conversion 関数),791
numeric,
MapForce 関数として (lang | logical 関数),842

O

Object,
reference, 1212
Object model,
overview, 1189
Object tree navigation,
Application, 1214, 1218
AppOutputLine, 1221, 1224
AppOutputLines, 1225, 1226
AppOutputLineSymbol, 1226, 1227
Component, 1228, 1232

Object tree navigation,
Components, 1234, 1235
Document, 1239, 1245
Documents, 1247, 1248
ErrorMarker, 1249, 1250
ErrorMarkers, 1251
MapForceView, 1253, 1255
Mapping, 1256, 1260
Mappings, 1260, 1261
Options, 1262, 1265
Project, 1269, 1273
ProjectItem, 1276, 1281
Obsolete, 1245
ODBC,
データ接続インターフェイスとして ,300
接続のセットアップ ,315
ODBC ドライバー ,
使用可能か確認する ,315
OLE DB,
データ接続インターフェイスとして ,300
OnDocumentClosed, 1239
OnDocumentOpened, 1213
OnProjectClosed, 1267
OnProjectOpened, 1213
OOXML,
デフォルト Excel 2007 フォーマットとして ,661
OpenDocument, 1217, 1248
OpenProject, 1218
Options, 1218, 1261
for code generation, 1245, 1264, 1265
for Java, 1243
Oracle,
XML 型フィールドから読み込む ,451
XML 型フィールドへ書き込む ,451
Oracle データベース ,
JDBC を使用しての接続 ,348
ODBC を使用しての接続 ,344
ORDER BY,
SQL where コンポーネント ,437
OS,
Altova 製品のための ,1438
Out of memory exceptions,
resolving, 1166
Output directory,
for code-generation files, 1262
for XSLT generated output, 1266
Output encoding,
default used, 1264
Overview,
of MapForce API, 1187

P

pad-string-left,
MapForce 関数として (lang | string 関数),848
pad-string-right,
MapForce 関数として (lang | string 関数),848
Parent, 1218, 1224, 1226, 1227, 1232, 1235, 1248, 1250, 1251, 1260, 1261, 1273
parent-context,
の定義 ,1461
parse-date,
MapForce 関数として (コア | conversion 関数),791
parse-dateTime,
MapForce 関数として (コア | conversion 関数),791
parse-number,
MapForce 関数として (コア | conversion 関数),793
parse-time,
MapForce 関数として (コア | conversion 関数),793
Path, 1245, 1274
PDF,
マッピング ドキュメント ,981
マッピングの出力のプレビュー ,977
pi,
MapForce 関数として (lang | math 関数),845
position,
MapForce 関数として (コア | sequence 関数),814
positive,
MapForce 関数として (lang | logical 関数),843
PostgreSQL,
ODBC を使用しての接続 ,349
直接接続する (ネイティブ),320
pow,
MapForce 関数として (lang | math 関数),845
Programming language,
enumerations for, 1285
Progress OpenEdge データベース ,
JDBC を使用しての接続 ,353
ODBC を使用しての接続 ,351
Project, 1266
creating new, 1217
file name, 1272
file name and path, 1270
on opening, 1213
opening, 1218
path with filename, 1274
saving, 1274

Project type,
enumerations for C#, 1286
for C#, 1264
for Java, 1265

Q

QName,
MapForce 関数として (lang | QName 関数),779
QName サポート,277
QName- as- string,
MapForce 関数として (lang | QName 関数),829
Quit, 1219

R

radians,
MapForce 関数として (lang | math 関数),845
random,
MapForce 関数として (lang | math 関数),845
RaptorXML Server,
変換の実行 ,946
remove- fileext,
MapForce 関数として (コア | file path 関数),795
remove- folder,
MapForce 関数として (コア | file path 関数),795
remove- timezone,
MapForce 関数として (lang | datetime 関数),840
repeat- string,
MapForce 関数として (lang | string 関数),849
replace,
MapForce 関数として (lang | string 関数),849
replace- fileext,
MapForce 関数として (コア | file path 関数),795
replicate- item,
MapForce 関数として (コア | sequence 関数),816
replicate- sequence,
MapForce 関数として (コア | sequence 関数),817
resolve- filepath,
MapForce 関数として (コア | file path 関数),795
resolve- uri,
MapForce 関数として (n xpath2 | anyURI 関数),853
reversefind- substring,
MapForce 関数として (lang | string 関数),849
right,
MapForce 関数として (lang | string 関数),849
right- trim,
MapForce 関数として (lang | string 関数),850
round,
MapForce 関数として (コア | math 関数),802
round- half- to- even,
MapForce 関数として (xpath2 | numeric 関数),859
round- precision,
MapForce 関数として (コア | math 関数),803
RSS フィード,
からのマッピング ,914
RTF,
マッピング ドキュメント,981
マッピングの出力のプレビュー ,977

S

SAP IDoc,
XML へのマッピング ,590
データのマッピング ,590
マッピング コンポーネントとして追加 ,590
SAP IDocs, 565
Save, 1245, 1274
SaveAs, 1246
Saved, 1246, 1274
Schema,
code generator, 1054
schemanativetype, 1169
second- from- datetime,
MapForce 関数として (lang | datetime 関数),840
second- from- duration,
MapForce 関数として (lang | datetime 関数),840
set- empty,
MapForce 関数として (コア | sequence 関数),817
set- null,
MapForce 関数として (db 関数),826
set- xsi- nil,
MapForce 関数として (コア | node 関数),804
ShowItemTypes, 1255
ShowLibraryFunctionHeader, 1256
shutdown,
of application, 1219
sin,
MapForce 関数として (lang | math 関数),846
skip- first- items,
MapForce 関数として (コア | sequence 関数),818
Soap,

Soap,
Web サービス ,885
Web サービス フォールト,885
フォールト,885
SOAP メッセージ ,
TTL (有効時間)の設定 ,908
ユーザー名トークンの追加 to, 908
SPL, 1168
code blocks, 1168
conditions, 1174
foreach, 1175
global objects, 1171
subroutines, 1176
using files, 1172
variables, 1170
Split once,
as FlexText function, 541
SPS,
ユーザー定義 スタイルシート,988
定義済みの スタイルシートfor documenting mappings, 985
SQL, 463
SQL スクリプトのインポート,464
SQL スクリプトをとして文字列ステートメントをエクスポートする , 464
スクリプトからロード,462
ステートメントの作成 ,463
ステートメントの実行 ,462,463
ステートメントの生成 ,462,463
SQL Server,
ADO を介しての接続 ,305
ADO.NET を使用して接続する ,309
XML 型フィールドから読み込む ,451
XML 型フィールドへ書き込む ,451
SQL WHERE,
ORDER BY, 437
コンポーネント - 挿入 ,437
SQL WHERE/ ORDER,
MapForce コンポーネントとして ,440
SQL エディター ,
コメントの挿入 ,464
テキストのコメントアウト,464
ブックマークの挿入 ,465
ブックマーク余白 ,465
作成 領域 ,465
削除 コメント,464
削除 ブックマーク ,465
削除 領域 ,465
使用 ブックマーク ,465
使用 領域 ,465
設定 - 一般 ,473
領域の挿入 ,465
SQLite,
XML からのマッピングデータ,321
XML ファイルを書き込む ,454
へのデータのマッピング ,210
以下からデータをマッピングする ,205
生成されたコード内でデータベースパスヲ絶対パスに変更する ,117
接続のセットアップ (Linux), 356
接続のセットアップ (Mac), 356
接続の設定 (Windows), 321
相対と絶対パスの使用 ,115
sqrt,
MapForce 関数として (lang | math 関数),846
starts- with,
MapForce 関数として (コア | string 関数),819
startup,
of application, 1219
static- node- annotation,
MapForce 関数として (コア | node 関数),805
static- node- name,
MapForce 関数として (コア | node 関数),805
Status, 1219, 1282
Store value,
FlexText 関数として ,557
string,
MapForce 関数として (コア | conversion 関数),794
MapForce 関数として (xpath2 | accessors ライブラリ),853
string- as- QName,
MapForce 関数として (lang | QName 関数),829
string- compare,
MapForce 関数として (lang | string 関数),850
string- compare- ignore- case,
MapForce 関数として (lang | string 関数),850
string- join,
MapForce 関数として (コア | aggregate 関数),784
string- length,
MapForce 関数として (コア | string 関数),819
Stylevision,
マッピングのために SPS スタイルシートを定義 ,988
substitute- missing,
MapForce 関数として (コア | sequence 関数),818
substitute- missing- with- xsi- nil,
MapForce 関数として (コア | node 関数),805
substitute- null,
MapForce 関数として (db 関数),826
substring,

substring,
MapForce 関数として (コア | string 関数),820
substring- after,
MapForce 関数として (コア | string 関数),820
substring- before,
MapForce 関数として (コア | string 関数),820
subtract,
MapForce 関数として (コア | math 関数),803
sum,
MapForce 関数として (コア | aggregate 関数),784
Sybase,
JDBC を使用しての接続 ,354
system- property,
MapForce 関数として (xslt | xslt 関数ライブラリ),864

T

tan,
MapForce 関数として (lang | math 関数),846
time- from- datetime,
MapForce 関数として (lang | datetime 関数),840
time- to- xlsx,
MapForce 関数として ,852
timezone,
MapForce 関数として (lang | datetime 関数),840
to- date,
MapForce 関数として (edifact 関数),827
to- datetime,
MapForce 関数として (edifact 関数),828
to- duration,
MapForce 関数として (edifact 関数),828
tokenize,
MapForce 関数として (コア | string 関数),820
tokenize- by- length,
MapForce 関数として (コア | string 関数),822
tokenize- regexp,
MapForce 関数として (コア | string 関数),824
to- time,
MapForce 関数として (edifact 関数),829
TRADACOMS, 565
XML に変換する ,602
マッピング コンポーネント,592
マッピング コンポーネントとして ,597
検証のルール ,598
構成 ファイル ,601
translate (コア | string 関数),
MapForce 関数として ,824
true,
MapForce 関数として (xpath2 | boolean 関数),854
Types,
built in, 1179
派生した型 - xsi:type, 275

U

UN/ EDIFACT, 565
XML にマッピング ,608
メッセージ構成のカスタム化 ,630
用語 ,567
unary- minus,
MapForce 関数として (lang | math 関数),846
Unicode,
code point 照合 ,170
特別文字の置き換え ,377
Unicode のサポート,
Altova 製品内で ,1439
unparsed- entity- uri,
MapForce 関数として (xslt | xslt 関数ライブラリ),864
uppercase,
MapForce 関数として (lang | string 関数),850
URI,
DTD 内 ,275
とQNames, 277
URL,
ファイルをコンポーネントとして追加 ,64

V

Value- Map,
ルックアップテーブル ,191
ルックアップテーブル - プロパティ,196
変更されていないデータをパス ,194
Variables,
in SPL, 1170
Version, 1216
View,
of MapForce, 1252
Visible, 1219
Visual Basic,
error handling, 1189
integration of MapForce, 1322
Visual Studio,

Visual Studio,
adding the MapForce ActiveX Controls to the toolbox, 1293
generating code for, 1167
generating mapping code for, 1059, 1063
Visual Studio のための MapForce プラグイン,
マッピングとプロジェクトとの作業,1001
共通のメニューと関数にアクセスする,1003
有効化,999
Visual Studio プラグイン,
として MapForce を実行する,998
VS .NET, 883
C# Web サービスのコンパイル,883
C# Web サービスの作成/デプロイ,883
Web サービス,883
作成 C#, 883

W

WADL, 902
Web services,
calling through HTTPS, 921
Web サービス,
HTTPS デジタル証明書を使用して署名,906
Java 内で作成,877
JSON にマッピング,911
MapForce からの呼び出し,888
MapForce コンポーネントとして,888
REST-スタイル 呼び出しの追加,911
RSS フィードからのマッピング,914
WS-Security の有効化,908
サンプル (WSDL-スタイル),918
タイムアウトの期間を構成する,902
デプロイ,877
プリエンプティブ資格の有効化,906
ユーザー名トークンの追加,908
基本 HTTP 認証資格を提供,906
実装,868
必要条件,868
例 (REST-スタイル),911
Web サービス (REST),
マッピングに追加,890
Web サービス (WSDL),
マッピングに追加,901
Web サービスプロジェクト,
作成,874
WebDAV Server,
ファイルをコンポーネントとして追加,64
weekday,
MapForce 関数として (lang | datetime 関数),841
weeknumber,
MapForce 関数として (lang | datetime 関数),841
WHERE,
SQL WHERE コンポーネント,437
WindowHandle, 1220
Windows,
Altova 製品のためのサポート,1438
executing mappings with Web service calls through HTTPS, 935
trusting server certificates on, 927
サーバー実行ファイルをデプロイ,355
Word,
マッピング ドキュメント,981
Word 2007+,
マッピングの出力のプレビュー,977
Wrapper classes,
in generated code, 1167
WSDL, 871
2.0 サポート,871
Web サービスをマッピングから呼び出す,918
サポートされるプロトコル,871

X

X12,
HIPAA, 579
X12 997 Functional Acknowledgement の生成,577
X12 999 Implementation Acknowledgement の生成,578
マージされたエントリ,624
XBRL タクソノミ パッケージ,
追加,684
xbrl-measure-currency,
MapForce 関数として,851
xbrl-measure-pure,
MapForce 関数として,851
xbrl-measure-shares,
MapForce 関数として,851
Xerces XML library,
generating code for, 1167
xlsx-to-date,
MapForce 関数として,852
xlsx-to-datetime,
MapForce 関数として,852
xlsx-to-time,
MapForce 関数として,853
XML,

XML,
CSV からのデータマッピング ,495
Excel 2007+ からのマッピング ,674
TRADACOMS から変換する ,602
UN/ EDIFACTXML からマッピング ,608
データベース フィールドへ書き込む ,454
マッピングターゲットとして ,495
署名 設定 ,294
XML から XML へ ,271
XML データ,
データベース フィールドから読み込む ,451
データベース フィールドへ書き込む ,451
XML パーサー ,
について ,1438
XML ファイル ,
Excel から生成する ,144
データベースレコードから生成する ,143
単一の XML ソースから生成する ,141
XML 出力 ,
インスタンス ファイル名の変更 ,271
エンコード設定の変更 ,271
スキーマの変更 ,271
作成 デジタル署名 ,271
XML 宣言 ,
出力からの抑制 ,271
XMLSpy command table, 1325
XQuery,
拡張関数 ,1421
生成されたコードのプレビュー ,80
追加 カスタム関数 ,758
XQuery プロセッサー ,
Altova 製品内で ,1438
xs: any (xs:anyAttribute), 283
xsi:nil,
XML インスタンス内の属性として ,278
xsi:type,
マッピング to 派生した型 ,275
XSLT,
code generation, 1242
options, 1266
カスタム関数の削除 ,755
カスタム関数の追加 ,755
テンプレート名前空間 ,755
拡張関数 ,1421
生成されたコードのプレビュー ,79
XSLT と XQuery のための Java 拡張関数 ,
Java 拡張関数を参照する ,1422
XSLT と XQuery のための拡張関数 ,1421
XSLT に対する MSXSL スクリプト, 1435
XSLT プロセッサー ,
Altova 製品内で ,1438
XSLT/ XQuery 内のスクリプト,
拡張関数で参照する ,1421

Y

year- from- datetime,
MapForce 関数として (ang | datetime 関数),841
year- from- duration,
MapForce 関数として (ang | datetime 関数),841

Z

Z to A,
sort コンポーネント,170
zip64mode,
enabling in the build.xml file, 1166
アイコン ,
データベース クエリのメッセージウィンドウ ,470
データベース クエリの結果ウィンドウ ,470
アイテム ,
不足している ,102
アプリケーション ワークフロー ,
使用 グローバルリソース ,964
インスタンス ,
ファイル参照をに変更する ,113
インターネットの使用 ,
Altova 製品内で ,1439
インパクトの分析 ,
SPS スタイルシート,985
インライン ,
関数とコードサイズ ,729
インライン / 標準 ,
ユーザー定義関数 ,729
インライン XBRL,
からデータをマッピングする ,683
エンコード,
CSV ファイル内の変更 ,502
フラットテキストファイルの変更 ,513
エンコード設定 ,
XML 出力 ,271
エンドユーザー使用許諾契約書 ,1440, 1442
エンベロープ ,
デジタル署名 ,294

オプション ,
テキストフォント - データベース クエリ ,475
結果表示 - データベース クエリ ,474
カスタム ,
XQuery 関数 ,758
カスタム ライブラリ ,
追加 ,763
キー ,
テキストファイル内のフィールド ,499
並べ替えキー ,170
グローバル リソース ,958
コンポーネントへ割り当て ,959
ツールバー ,957
データベースとして ,970
デフォルト構成 ,958
フォルダーとして ,962
プロパティ ,973
リソースファイルの定義 ,958
ワークフロー ,964
ワークフローの開始 ,968
構成のコピー ,958
定義ファイル ,957
有効化 ,961
コード ,
インライン 関数 &コードのサイズ ,729
コードポイント ,
照合 ,170
コード生成設定 ,
プロジェクト全体のためにグローバルに定義する ,108
プロジェクト内のフォルダーのために定義する ,109
コネクタ ,
の定義 ,1455
処理された値の履歴の確認 ,258
全てコピー ,126
コメント ,
ターゲット ファイルに追加 ,281
コンテキスト ウィンドウ ,
について ,253,260
コンポーネント ,
sort data, 170
アプリケーション メニューとして ,1038
グローバル リソースの割り当て ,959
の定義 ,1455
マッピングに追加 ,63
概要 ,84
検索 ,85
削除されたアイテム ,102
処理シーケンス ,237
整列 ,86
設定の変更 ,87
コンポーネントへ グローバル リソースを ,959
サンプル ,
再帰的 ユーザー定義 マッピング ,748
シーケンス ,
処理コンポーネントの ,237
システム DSN,
セットアップ ,315
スキーマ ,
generating for an XML ファイル ,271
と XML マッピング ,271
ファイル参照をに変更する ,113
再帰的要素 ,748
スタイル シート ,
ドキュメントのために定義 ,988
セクション ,
CDATA, 282
セパレーター ,
EDI, 567
ソース コンポーネント ,
の定義 ,1462
ソース優先 ,
- 混在コンテンツマッピング ,119
ソース優先接続 ,
通常 (ターゲット優先)接続に対して ,125
ソフトウェア製品 ライセンス ,1442
ターゲット ,
FlexText コンポーネント ,534
ターゲット コンポーネント ,
の定義 ,1463
ターゲット優先 マッピング ,119
ターゲット優先接続 ,
ソース優先接続に対して ,125
チャート関数 ,
サンプル ,1419
チャートデータ構造 ,1414
リスト ,1410
ツール ,
アプリケーション メニューとして ,1046
ツールバー ,
グローバル リソース ,957
データオーバーレイ ,
について ,253
データ ストリーミング ,
定義 ,66
データのオーダー ,
並べ替えコンポーネント ,170
データのパススルー ,
value-map をスルー ,194

データの統合 ,
XML ファイルのマージ ,287
データの保持 ,194
value-map の使用 ,194
value-map をスルー ,194
データベース ,
XML スキーマをフィールドに割り当てる ,451
XML ファイルを書き込む ,454
クエリ ,460
グローバルリソースとして ,970
シーケンスになった一意の値を生成 ,379
データを複数にリンクされているテーブルに挿入する ,387
テーブルデータの更新 ,392
テーブルデータの削除 ,407
テーブルにデータを挿入する ,384
テーブルの更新と挿入 ,395
トランザクションロールバック ,413
にデータをマージする ,395
に対してマッピングを実行する ,376
バルク挿入の使用 ,415
マッピングに追加 ,357
複数の XML ファイルから生成する ,143
データベース クエリ ,
SQL の生成 ,466 ,470
コメントアウトテキスト ,464
テキストフォントオプション ,475
フィルタリング テーブル ,467
ブックマーク ,465
結果表示 オプション ,474
領域 ,465
データベース リレーションシップ ,
マッピング内で定義 ,371
マッピング内に保持 ,363
データベースオブジェクト ,
フィルタリング ,360
マッピングから削除 ,360
マッピングへ追加 ,360
データベースドライバー ,
概要 ,302
データベースへのアクセス ,
IF 条件をベースにした更新 ,506
データベース接続 ,
ウィザードの開始 ,301
グローバルリソースからの再使用 ,325
セットアップサンプル ,326
設定 ,300
テーブル ,
ルックアップテーブル ,191
テーブルデータ ,
並べ替え ,170
テキスト ,
ファイル - キーフィールドの定義 ,499
マッピング テキストファイル ,495
テキストファイル ,
エンコードの設定 ,513
からのデータのプレビュー ,513
ソース コンポーネントとして ,513
ターゲットコンポーネントとして ,513
データのマッピング ,506
フィールドの追加または削除 ,513
固定されたフィールドサイズの設定 ,506
充填文字の設定 ,506
デジタル署名 ,
XML 出力の作成 ,271
アクティブ化 ,290
設定 ,294
デタッチ済み ,
デジタル署名 ,294
デバッガーポジション ,
の現在の値野確認 ,258
デバッグ ,
ステップバイステップ ,248
について ,253
のための準備 ,251
ブレークポイントと ,248
開始 ,252
制限 ,248
設定 ,268
停止 ,252
デフォルト ,
構成 - グローバルリソース ,958
入力値 ,734
ドキュメント ,
SPS スタイルシートを定義 ,988
ドキュメント化 ,
マッピング ,981
ドキュメント概要 ,
SPS スタイルシート ,985
ナビゲート ,
ブックマーク ,465
により使用された関数 ,985
ネストされた ,
ユーザー定義関数 ,734
ノード ,
FlexText 関数として ,549
ノード名 ,
データをマッピング ,215
パーサー ,

パーサー ,
Altova 製品に内蔵の ,1438
パスワード ,
デジタル署名 ,294
バッチ ,
自動化された 処理 ,939
パラメーター ,734
出力 ,734
任意の ,734
バリデーター ,
Altova 製品内で ,1438
ファイル ,958
Excel から複数のファイルを生成する ,144
アプリケーションメニューとして ,1031
グローバルリソース ,958
コンポーネントとしてのボタン ,87
コンポーネントのボタンとして ,136
リソース 構成の追加 ,958
ファイル DSN,
セットアップ ,315
ファイル /文字列 ,
コンポーネントとしてのボタン ,87
コンポーネントのボタンとして ,136
ファイル : (デフォルト),
ルートノードの名前として ,136
ファイル : <dynamic>,
ルートノードの名前として ,136
ファイルパス ,
ファイルベースのデータベースの ,115
生成されたコード内で ,117
相対と絶対 ,113,117
破損した参照の修正 ,117
ファイル名 ,
マッピング入力パラメーターとして提供する ,140
フィールド ,
テキストファイル内のキー ,499
フィルター ,
XML ファイルのマージ ,287
オンラインブラウザー ,467
データベース オブジェクト,467
データベース内 ,440
マッピングに追加 ,175
全てコピー コネクタ,126
フィルタリング ,
コンポーネントからのデータ,175
データベーステーブル ,175
マッピング上のデータベースオブジェクト,360
フォールト ,
Web サービス ,885
フォルダー ,
グローバルリソースとして ,962
レイアウト - データベース クエリ,466
ブックマーク ,
ナビゲート,465
ブックマーク余白 ,465
削除 ,465
挿入 ,465
ブラウザー ,
データベース クエリ,466
フィルターの適用 ,467
フィルタリング ,467
ブラウザービュー ,
SQL ステートメントの生成 ,468
コンテキストメニュー オプション ,468
フラットファイル ,
マッピング ,495
プラットフォーム ,
Altova 製品のための ,1438
ブレークポイント ,
について ,253
削除 ,256
追加 ,256
ブレークポイントウィンドウ ,
について ,253,261
プロジェクト ,
アプリケーションメニューとして ,1036
開く,107
検索 ,107
作成 ,107
閉じる ,107
ブロックコメント ,464
プロトコル ,
サポートされる WSDL, 871
プロパティ ,
value map テーブル ,196
ヘルプ ,
アプリケーションメニューとして ,1048
マークされたアイテム ,
不足しているアイテム ,102
マージ ,
XML ファイル ,287
マージされたエントリ ,624
マッピング ,
ソース優先 - 混在コンテンツ ,119
デバッグ ,248
ドキュメント化 ,981
の定義 ,1459
フラットファイルフォーマット,495

マッピング ,
型優先 ,126
検証 ,68
作成 ,63
自動化された 処理 ,939
処理シーケンス ,237
定義済みの SPS スタイルシートfor ドキュメント,985
マッピング 出力 ,
として複数のファイルを生成する ,136,139
マッピングメソッド,
ターゲット優先 ,119
標準 ,119
標準 / 混在 / 全てコピー ,119
マッピング入力 ,
としてカスタムファイル名を提供する ,140
として複数のファイルを提供する ,136,138,139
メッセージ ,
ウィンドウ - データベース クエリ,470
データベース クエリ内のアイコン ,470
メモリ不足エラー ,
トラブルシューティング ,66
メモリ要件 ,1438
ユーザー DSN,
セットアップ ,315
ユーザー定義 ,734
bool = false の場合の出力 ,734
ネストされた関数 ,734
ルックアップ 関数 ,731
関数 - インライン / 標準 ,729
関数 - 標準 ,731
関数 - 複合型 ,739,743
複合型 出力 ,744
複合型入力 ,739
ユーザー定義関数 ,
インポート,722
パラメーターの順序に影響する ,722
開く,722
再帰的 ,749
再使用 ,722
作成 ,722
削除 ,722
ライセンス ,1442
情報 ,1440
ライセンス計測 ,
Altova 製品にて ,1441
ライブラリ ,
カスタムを追加 ,763
の自動ロード,763
追加 XQuery 関数 ,758
ライブラリウィンドウ ,
内で関数を検索 ,719
ライブラリファイル ,
mff, 763
ラインコメント,464
リソース ,
グローバルリソース プロパティ,973
データベースとして ,970
フォルダー ,962
ルックアップテーブル ,
value map テーブル ,191
プロパティ,196
レイアウト,
ブラウザー ,466
レファレンス ,1030
ローカル照合 ,170
ワークシート,
XML 出力 ファイルへ ,144
ワークフロー ,
グローバルリソースの開始 ,968
使用 グローバルリソース ,964
ワイルドカード,
xs:any - xs:anyAtrribute, 283
引用文字 ,
CSV ファイル内 ,502
階層構造 ,
テキストファイルから ,499
外部キー ,
データベースマッピング内 ,363
データベースマッピング内の ,371
割り当て ,959
関数 ,729
アクティブなマッピングで発生をビュー ,719
アプリケーションメニューとして ,1041
インライン ,729
から関数を削除 ,719
にパラメーターを追加 ,719
ネストされたユーザー定義 ,734
の詳細をビュー ,719
マッピングコンポーネントとして追加 ,719
ユーザー定義ルックアップ関数 ,731
ユーザー定義関数 ,749
ライブラリ内出検索 ,719
引数のデータ型のビュー ,719
追加 カスタム XQuery, 758
標準 ユーザー定義関数 ,731
複合型 - インライン ,729
技術データ,1438
疑問符 ,

疑問符 ,
不足しているアイテム ,102
区切り文字 ,
CSV ファイル内の変更 ,502
フラットテキストファイルの変更 ,513
空のフィールド ,
CSV ファイル内 ,502
フラットテキストファイル内 ,513
繰り返される分割 ,
FlexText 関数として ,535
using the "fixed length" mode, 536
区切られた (フロート) モードの使用 ,537
区切られた (行の開始) モードの使用 ,539
区切られた (行ベース) モードの使用 ,538
型優先 ,
接続 ,126
結果 ,
ウィンドウ - データベース クエリ ,470
データベース クエリ内のアイコン ,470
検索 ,
iマッピング コンポーネント内のアイテム ,85
プロジェクトウィンドウ内のファイル ,107
ライブラリウィンドウ内の関数 ,719
検証する ,
デザインのマッピング ,68
マッピング 出力 ,69
呼び出しグラフ ,
SPS スタイルシート ,985
固定された ,
固定長ファイル - マッピング ,495
構成 ,958
mff ファイル ,763
SQL エディター 設定 ,473
グローバルリソースに追加 ,958
既存のコピー ,958
切り替え - グローバルリソース ,961
行 ,
Excel から ,144
テキストファイルからのマッピング ,499
混在 ,119
コンテンツマッピング ,119
コンテンツマッピング サンプル ,124
コンテンツマッピングのメソッド ,119
ソース優先 マッピング ,119
再帰的 ,
ユーザー定義 マッピング ,748
ユーザー定義関数 ,749
関数内の呼び出し ,729
作成 ,
regions, 465
削除 ,126
regions, 465
コメント ,464
ブックマーク ,465
ブロックコメント ,464
ラインコメント ,464
削除 - 不足しているアイテム ,102
全てコピー 接続 ,126
使用 ,
グローバルリソース ,961
試用期間 ,
Altova ソフトウェア製品の試用 ,1440 ,1441
自動 ,
ライブラリのロード ,763
自動化された ,
処理 ,939
主キー ,
データベース コンポーネントのために生成 ,379
データベースマッピング内 ,363
データベースマッピング内の ,371
データベース挿入アクション中に生成する ,384
出力 ,734
bool = false の場合の ,734
アプリケーションメニューとして ,1042
パラメーター ,734
プレビュー ,70
検証 ,69
保存 ,70
出力 コンポーネント ,
の定義 ,1460
順序 ,
コンポーネントが処理されます ,237
処理 ,
マッピングの自動化 ,939
処理シーケンス ,
マッピング内のコンポーネント ,237
処理命令 ,
ターゲットファイルに追加 ,281
処理命令とコメント ,
マッピング ,120
署名 ,
デジタル署名 ,290
設定 ,294
照合 ,
locale 照合 ,170
sort コンポーネント ,170
unicode code point, 170
証明書 ,

証明書 ,
デジタル署名 ,294
正規表現 ,
"match-pattern" 関数のパラメーターとして ,776
"tokenize-regexp" 関数のパラメーターとして ,776
FlexText トコンポーネントの分割 ,559
FlexText 内での条件の変更の使用 ,560
MapForce FlexText 内の ,558
生成 ,
コード &インライン 関数 ,729
デジタル署名 ,290
生成されたコード ,
から例外をスロー ,199
生成されたコード内のパス ,
絶対パスに設定する ,81
切り替え ,
FlexText 関数として ,546
構成 - グローバルリソース ,961
接続 ,
アプリケーションメニューとして ,1040
の定義 ,1455
ルート要素の変更を保存する ,98
異なるコンポーネントへ移動する ,98
型優先 ,126
折りたたみ ,
regions, 465
設定 ,
テキストフォント - データベース クエリ ,475
デジタル署名 ,294
結果表示 - データベース クエリ ,474
先頭の 0 ,
変換中に保持する ,601
選択 ,
テーブル データ - データベース クエリ ,470
全てコピー ,
コネクタ ,126
とフィルター ,126
マッピングメソッド ,119
解決 / 削除 コネクタ ,126
挿入 ,
regions, 465
SQL WHERE, 437
アプリケーションメニューとして ,1034
コメント ,464
ブックマーク ,465
ブロックコメント ,464
ラインコメント ,464
単一のターゲット ,
複数のソース ,287
単純型 ,
sorting, 170
値 ,
デフォルト ,734
値ウィンドウ ,
コンテキストタブ ,258
について ,253 ,258
関連したタブ ,258
履歴タブ ,258
著作権情報 ,1440
追加 ,
MapForce 関数として (コア | math 関数),801
カスタムライブラリ ,763
グローバルリソース ファイル ,958
定義ファイル ,
globalresource.xml, 957
展開 ,
regions, 465
内蔵のエンジン ,
使用 ,67
定義 ,67
入力 ,734
デフォルト value, 734
任意の パラメーター ,734
入力コンポーネント ,
の定義 ,1457
入力の複製 ,40
追加 ,1038
任意の ,
入力パラメーター ,734
派生した型 ,
マッピング to/ from, 275
背景情報 ,1438
配布 ,
Altova ソフトウェア製品の配布 ,1440 ,1441
Altova のソフトウェア製品 ,1440
標準 ,
マッピングメソッド ,119
標準的な XML,
デジタル署名 ,294
表示 ,
アプリケーションメニューとして ,1044
評価期間 ,
Altova のソフトウェア製品 ,1440
不足しているアイテム ,102
複合コンテンツ ,
マッピング ,125
複合型 ,
ユーザー定義 複合型 出力 ,744

複合型 ,
ユーザー定義 複合型入力 ,739
ユーザー定義関数 ,739,743
関数 - インライン ,729
並べ替え ,170
複数の XML ファイル ,
Excel から ,144
複数のソース ,
単一のターゲットに ,287
複数の連続した要素 ,
Edifact - X12, 624
文字の重点 ,
フラットテキストファイル内 ,513
文字列 ,
からデータを解析する ,203
データのシリアル化 ,203
データの解析 ,203
データを以下から解析する ,205
へのデータのシリアル化 ,203,210
並べ替え ,
データベース内 ,440
テーブル データベース クエリ,466
結果ウィンドウ内のデータ,470
結果ウィンドウ内の列アイコン ,470
並べ替えコンポーネント,170
並べ替えの順序 ,
変更 ,170
並べ替キー ,
並べ替えコンポーネント,170
変換 ,
RaptorXML Server, 946
入力 データ - value map, 191
変換の結果 ,
グローバルリソース ,964
変換言語 ,
選択 ,67
変更 ,
構成 - グローバルリソース ,961
変数 ,
のスコープの変更 ,164
マッピングに追加 ,161
使用例 ,166,167,168
始めに ,160
編集 ,
アプリケーションメニューとして ,1033
法的な情報 ,1440
無視する ,
FlexText 関数として ,550
名前空間 ,
カスタムの宣言 ,288
とワイルドカード (xs:any), 283
名前空間 URI,
DTD, 275
とQNames, 277
優先コンテキスト,
関数の設定 ,239
用語 ,
EDIFACT, 567
要素 ,
XML スキーマ内の再帰的要素 ,748
領域 ,
作成 ,465
削除 ,465
折りたたみ ,465
挿入 ,465
展開 ,465
例外 ,
Web サービス フォールト,885
追加 ,199